

AD-A195 851

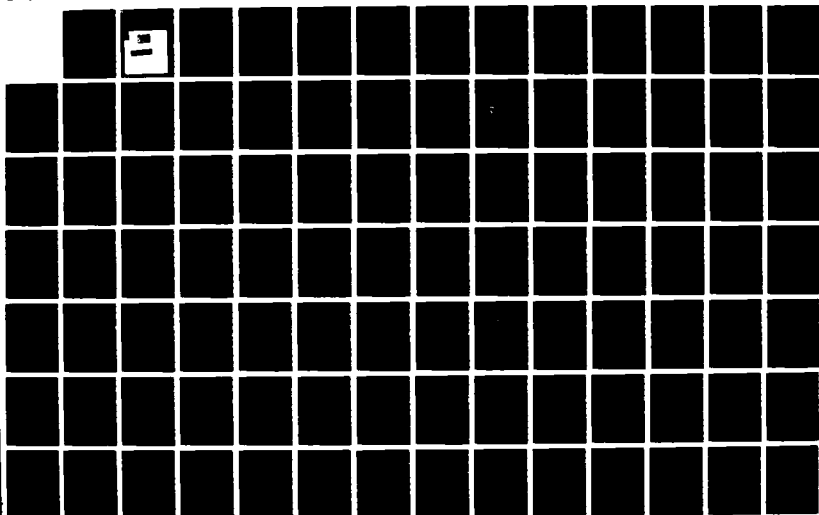
KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS
DEVELOPMENT METHODOLOGIES. (U) MASSACHUSETTS INST OF
TECH CAMBRIDGE A GUPTA ET AL. DEC 87 MIT-KBIISE-2
DTRS57-85-C-00003

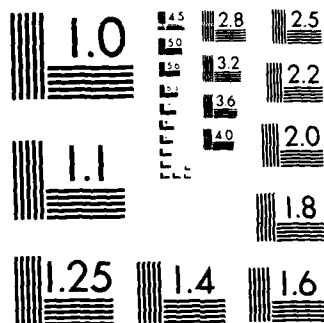
1/4

UNCLASSIFIED

F/B 12/7

ML





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

DTIC FILE COPY



Massachusetts
Institute of
Technology

Knowledge-Based
Integrated Information
Systems Engineering
(KBIIE) Project

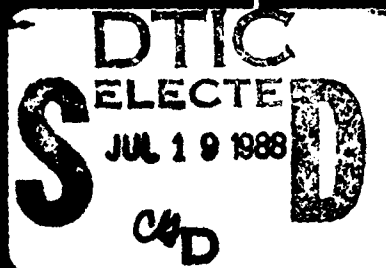
Volume 2

Knowledge-Based Integrated Information Systems Development Methodologies Plan

2

AD-A195 851

Amar Gupta
Stuart Madnick
SERIES EDITORS



DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER KBIISE-2	2. GOVT ACCESSION NO. A155 851	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Knowledge-Based Integrated Information Systems Development Methodologies Plan		5. TYPE OF REPORT & PERIOD COVERED Part of Final Report; Sept. 86 - Jan. 88
		6. PERFORMING ORG. REPORT NUMBER KBIISE-2
7. AUTHOR(s) Amar Gupta and Stuart Madnick (editors)		8. CONTRACT OR GRANT NUMBER(s) DTRS57-85-C-00083
9. PERFORMING ORGANIZATION NAME AND ADDRESS Massachusetts Institute of Technology Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Transportation Systems Center, Broadway, MA 02142 (In conjunction with U.S. Air Force)		12. REPORT DATE December 1987
		13. NUMBER OF PAGES 342 pages
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Transportation Systems Center, Broadway, MA 02142		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for Public Release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES This volume is one of eight volumes prepared by M.I.T. for Department of Transportation and Department of Defense (U.S. Air Force).		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Integration, knowledge, databases, systems engineering, methodologies, information modeling, heterogeneous database systems		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This volume describes the Integrated Information System Evolution Environment (IISEE) which is a structured approach for the strategic planning, tactical planning, requirements definition, design, construction, implementation, and maintenance of large scale distributed, heterogeneous integrated information systems. This approach is designed to enable system developers in the Air Force and its contractor community to evolve their current systems into a unified framework.		

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

The overall goal of the IISEE thrust is to establish a structured systems development approach consisting of methodologies, methods, tools, techniques, and practices which can improve productivity of participants, and reduce cost and time required for integrated information systems development. The IISEE strategy has been designed keeping in view the objectives of the Integrated Design Support System (IDS) Program of the Air Force. The latter program encompasses all issues relating to capture, management, and communication of product technical data from its initial creation in design, through realization in manufacture, to logistics operations.

The work described in this volume has been jointly conducted by several contractors and subcontractors. The individual inputs have been edited and consolidated by a research team led by Richard J. Mayer of the Knowledge Based Systems Laboratory at Texas A&M University.

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS DEVELOPMENT METHODOLOGIES PLAN

Amar Gupta
Stuart Madnick

Series Editors



Accession For	
NTIS	✓
DTIC	
Unannounced	
Justification	
By	
Date	
A-1	

Knowledge-Based Integrated Information Systems
Engineering (KBIISE) Report: Volume 2

Massachusetts Institute of Technology

Copyright © 1987 by
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
77 Massachusetts Avenue
Cambridge, MA 02139
All rights reserved.

This book is one of eight volumes published by M.I.T. as part of the Knowledge-Based Integrated Information Systems Engineering (KBIISE) Report. The contents of these eight volumes reflect the work performed by M.I.T. employees and students, as well as by a number of contractors and subcontractors. All requests for permission for photocopying and distribution of these volumes and individuals parts should be directed to the series' editors at M.I.T.

KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS DEVELOPMENT METHODOLOGIES PLAN

About This Volume

This volume describes the Integrated Information System Evolution Environment (IISEE) which is a structured approach for the strategic planning, tactical planning, requirements definition, design, construction, implementation, and maintenance of large scale distributed, heterogeneous integrated information systems. This approach is designed to enable system developers in the Air Force and its contractor community to evolve their current systems into a unified framework.

The overall goal of the IISEE thrust is to establish a structured systems development approach consisting of methodologies, methods, tools, techniques, and practices which can improve productivity of participants, and reduce cost and time required for integrated information systems development. The IISEE strategy has been designed keeping in view the objectives of the Integrated Design Support System (IDS) Program of the Air Force. The latter program encompasses all issues relating to capture, management, and communication of product technical data from its initial creation in design, through realization in manufacture, to logistics operations.

The work described in this volume has been jointly conducted by several contractors and subcontractors. The individual inputs have been edited and consolidated by a research team led by Richard J. Mayer of the Knowledge Based Systems Laboratory at Texas A&M University.

Table of Contents

	Page
SERIES EDITORS' NOTE	1
KNOWLEDGE BASED INTEGRATED INFORMATION SYSTEMS DEVELOPMENT METHODOLOGIES PLAN (Technical Report #3)	5

Knowledge-Based Integrated Information System Engineering Project: Volume 2
Amar Gupta and Stuart E. Madnick, Editors
Copyright © Massachusetts Institute of Technology, 1987.

DEDICATED
TO
THE
NEXT
GENERATION
OF
PROFESSIONALS

SERIES EDITORS' NOTE

This book is one of eight volumes published by MIT as part of the Knowledge-Based Integrated Information Systems Engineering Project (KBIISE). In order to appreciate the papers in this book, it is necessary to be aware about the theme of the KBIISE project, its major objectives, and the different documents that summarize the research accomplishments to date.

Goal

The primary goal of the KBIISE project is to integrate islands of disparate information systems that characterize virtually all large organizations. The number and the size of these islands has grown over years and decades as organizations have invested in an increasing number of computer systems to support their growing reliance on computerized data. This has made the problem of integration more pronounced, complex, and challenging.

The need for multiple systems in large organizations is dictated by a combination of technical reasons (such as the desired level of processing power and the amount of storage space), organizational reasons (such as each department obtaining its own computer based on its function), and strategic reasons (such as the level of reliability, connectivity, and backup capabilities). Further, underlying trends in the information technology area have led to a situation where most organizations now depend on a portfolio of information processing machines, ranging from mainframes to minicomputers and from general purpose workstations to sophisticated CAD/CAM systems, to support their computational requirements. The tremendous diversity and the large size of the different systems make it difficult to integrate these systems.

Key Participants

The above problem is becoming increasingly evident in all large government agencies and in large development programs. In the fall of 1986, the U.S. Air Force (USAF) and the Transportation Systems Center (TSC) of the U.S. Department of Transportation approached M.I.T. to conduct and to coordinate research activity in this area in order "to develop the framework for a comprehensive methodology for large scale distributed, heterogeneous information systems which will provide: (i) the necessary structure and standards for an evolving top down global framework; (ii) simultaneous bottom up systems development; and (iii) migratory paths for existing systems."

Both USAF and TSC provided sustained assistance to members of our research team. In addition, Citibank and IBM provided some funds for research in very specific areas. One advantage of our corporate links was the opportunity to analyze and to generate case studies of actual decentralized organizational environments.

The research sponsors and MIT agreed that in order to deal with the heterogeneity issue in a meaningful way, it was important that a critical mass of influential individuals participate in the development of solutions. Only through widespread discussion and acceptance of a proposed strategy would it become feasible to deal with the major problems. For these reasons, a Technical Advisory Panel (TAP) was constituted. Nominees to the TAP included experts from academic and research organizations, government agencies, computer companies, and other corporations. In addition, several subcontractors, the primary one being Texas A&M University, provided assistance in specific areas.

Technical Outputs

The scope of the work included (i) technical issues; (ii) organizational issues; and (iii) strategic issues. On the basis of exploratory research efforts in all these areas, 24 technical reports were prepared. Eighteen of these reports were generated by MIT research personnel, and their respective areas of investigation are summarized in the figure on the opposite page.

The five technical reports, not represented in the figure, are as follows:

- #1. Summary.
- #2. Record of discussions held at the first meeting of the Technical Advisory Panel (TAP) on February 17, 1987.
- #3. Consolidated report submitted by Texas A&M University.
- #21. Annotated Bibliography.
- #23. Record of discussions held at the second meeting of the Technical Advisory Panel (TAP) on May 21 and 22, 1987.
- #24. Contributions received from members of the TAP highlighting their views on various aspects of the problem.

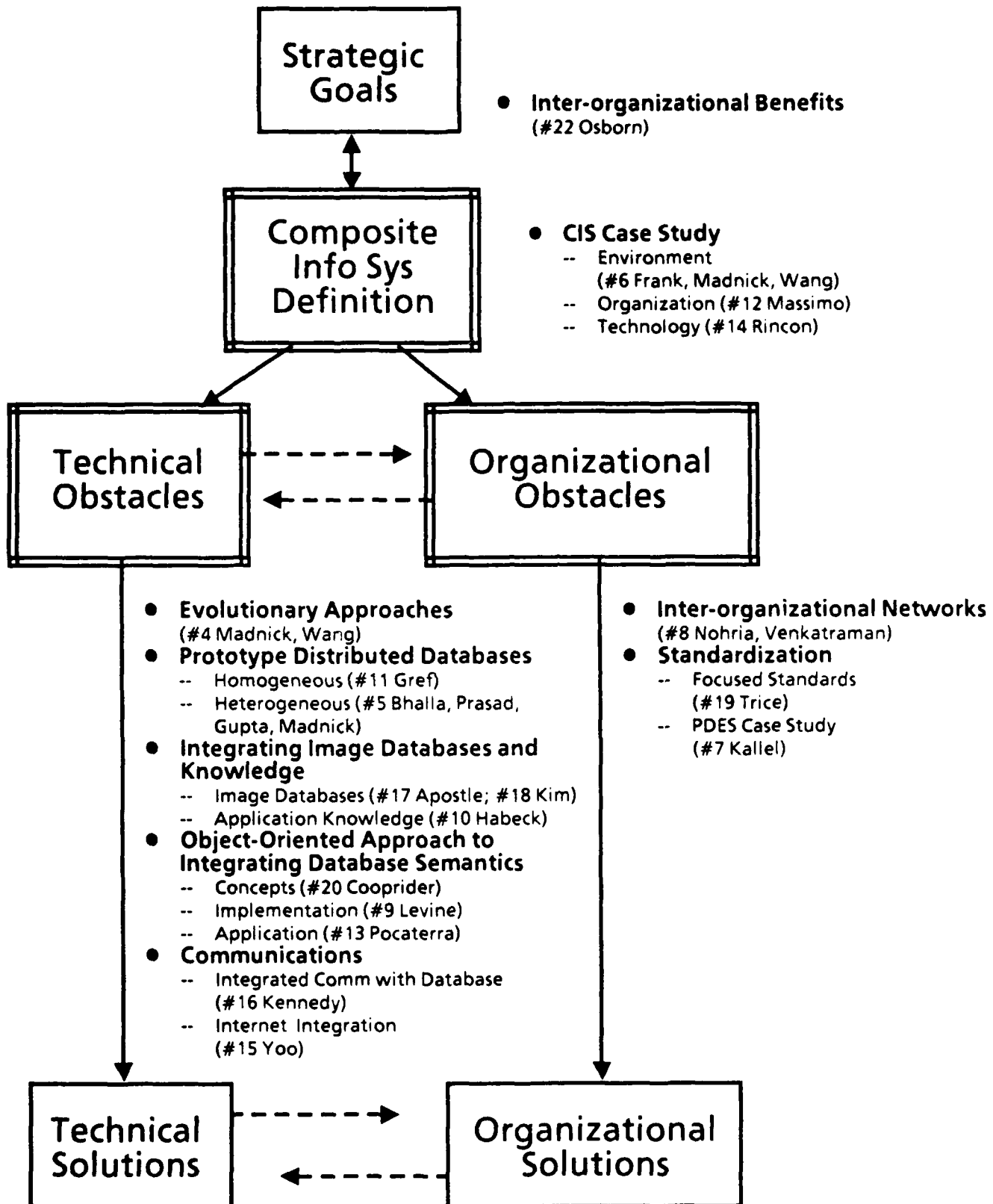
All the 24 technical reports have been edited and reorganized as an eight-volume set. The titles of the different volumes are as under:

- 1. KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS ENGINEERING-HIGHLIGHTS AND BIBLIOGRAPHY
- 2. KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS DEVELOPMENT METHODOLOGIES PLAN
- 3. INTEGRATING DISTRIBUTED HOMOGENEOUS AND HETEROGENEOUS DATABASES - PROTOTYPES
- 4. OBJECT-ORIENTED APPROACH TO INTEGRATING DATABASE SEMANTICS
- 5. INTEGRATING IMAGES, APPLICATIONS, AND COMMUNICATIONS NETWORKS
- 6. STRATEGIC, ORGANIZATIONAL, AND STANDARDIZATION ASPECTS OF INTEGRATED INFORMATION SYSTEMS
- 7. INTEGRATING INFORMATION SYSTEMS IN A MAJOR DECENTRALIZED INTERNATIONAL ORGANIZATION
- 8. TECHNICAL OPINIONS REGARDING KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS ENGINEERING

Volume 2 contains the report submitted by Texas A&M and Volume 8 highlights the views of members of the TAP. Activities described in the other 6 volumes have been conducted at MIT.

EXPLORATORY RESEARCH EFFORTS

3



Acknowledgments

Funds for this project have been provided by U.S. Air Force, U.S. Department of Transportation (Contract Number DTRS57-85-C-00083), IBM, and Citibank. We thank all these organizations and their representatives for their support. In particular, we are indebted to Major Paul Condit of U.S. Air Force for his initiative in sponsoring this project, to Dr. Frank Hassler, Bud Giangrande, and Bob Berk of the Transportation Systems Center (TSC) for their support and assistance, to Professor Joseph Sussman, Director, Center for Transportation Studies (CTS) at MIT for his help and encouragement, and to all the individuals whose results have been published in this book.

We would welcome receiving feedback from readers of this book.

Amar Gupta and S.E. Madnick
Massachusetts Institute of Technology
Cambridge, Massachusetts.

KNOWLEDGE BASED INTEGRATED INFORMATION SYSTEMS DEVELOPMENT METHODOLOGIES PLAN

**RICHARD J. MAYER, PATRICIA J. FRIEL, PAULA S. D. MAYER,
ALVIN J. UNDERBRINK, JR., CHRIS MENZEL, GUY BAILEY,
DON PHILLIPS, MAC LIVELY, LELAND BLANK, TOM CULLINANE,
SAM NUSINOW, GREG NUSINOW, STU COLEMAN, TIM RAMEY,
ROBERT BROWN, PAUL THOMPSON, FRANKLIN SKIPPER,
BILL KNAPPENBERGER, AND KEN MORRISON**

The key to making an effective Integrated Information System Evolution Environment (IISEE) is to develop a common core of methods which are modular and which interlock in a consistent fashion to cover the entire life cycle. Surrounding this cover, a set of special techniques must be developed for dealing with the special characteristics of the organization, the problem area, and the application technology.

The Integrated Design Support System (IDS) Program is focused on the capture, management, and communication of product technical data from its initial creation in design, its realization in manufacture, through logistics operations. The system encompasses several hundreds of contractors, and many millions of line of program code. The enormity of the task is somewhat mitigated by the fact that only the critical product data needs to be controlled.

The approach taken in IDS is to implement three schema architecture utilizing commercially available software mechanisms to the maximum extent possible. Enterprise information control is accomplished via the establishment of a "conceptual schema" and an executive control system. The conceptual schema contains a definition of the enterprise information management and control rules. The executive control system validates each transaction against these rules.

Two models of the system development process have been developed. The Information System Evolution Process model (ISEP-O) is aimed at systematically describing how to build and use Information Systems Integration shells, like an IDS shell. The Integrated Design System/System Development Process model (IDES/SDP-O) is designed to help describe the process of assimilation, customization, and utilization of an IDS shell. The IDDES/SDP-O model has been developed down to the second/third level of decomposition and, in places, extended further with a very detailed node index. These two models can help large organizations reduce the complexities involved in integrating multiple large distributed databases together.

Chapter 1

Management Summary

1.1 Introduction

The Air Force Integrated Design Support System (IDS) represents a major evolutionary step in information resource management. The evolution from stand alone applications to integrated information environments has caused the recognition of a need to provide a means of acquiring and using this new technology. An integrated methodology to support the major roles and activities of these new systems is required. The methodology must include specific methods, automated tools, integrating frameworks (how to procedures), and automated environments. By major roles we mean management, engineering, construction, and maintenance. By activities we include planning, definition, design, realization and maintenance. The term "new systems" covers the information integration utilities themselves, the new applications developed under these frameworks, and the legacy systems which must be interfaced to the new order. This project was initiated with the objective of uncovering what the current industry practitioners (those attempting to build and implement information integrated systems) needs were relative to such an integrated methodology. These needs were then translated into specific requirements. In many cases these requirements represent shortfalls in the current "applications engineering" based methods and tools. Based on these requirements and an understanding of the current state of the art in software development methods and tools we established a concept for an *Integrated Information System Evolution Environment* (IISEE). A program for the development of this concept from the current base of methods and tools was established including project descriptions, budgets, and timings. These results hopefully will serve as a basis for the establishment of a research and development program to produce the technology needed for an organization (commercial or government) to implement and evolve an information integrated solution to its current and future data needs.

1.2 Background

Establishment of a complete System Development Methodology (SDM) is in itself a complex system development process. In fact, several effective analogies can be drawn between a system development methodology and developing a large software system. Just as there are features and capabilities of a large software system which only a few (possibly only one) users need, there will be capabilities of an SDM which are only needed in a particular application development arena. Thus, an information system developer working in the implementation of a factory control system (within a three schema architecture concept like IDS¹) will have slightly different methodological needs than a developer of an engineering data control system.

The key to making an effective Integrated Information System Evolution Environment (IISSE) for IDS is to develop a common core of methods which are modular and which interlock in a consistent fashion to cover the entire life cycle. Surrounding this core, there must be a set of special techniques for dealing with the peculiarities of the users organization, a specific problem area, or an application technology. Thus, when a particular person approaches the IISSE he will find both the common core which will insure that the results of his development efforts will be information integratable with the IDS, and specific methods which allow him to deal with the particular problems unique to his own development efforts.

Almost all methods in widespread use today have evolved from engineering practice. In fact, a method can be viewed as a formalization of a commonly used (successful) engineering, management, production or support technique. Formalism is important because it allows the transition of the technique to other application areas; it allows one to define the limitations of the application of a technique, and it facilitates the interface to other techniques. In the early books on structured programming, for example, a question often posed was "What is the difference between what an "expert" programmer does and what the methodology suggests?" The conclusion was always that there is no difference. Rather the method was supposed to assist in bringing all the programmers up to the level of quality and productivity of the expert programmer.

Another case in point is to remember that structure charting and data flow diagramming (which Constantine developed and Yourdon has popularized) were developed by studying over 100 business application systems at IBM. Larry Constantine interviewed the developers and examined the maintenance and modification histories of the applications. From this base he distilled a methodology which he felt captured the "best practice" from this experience. The effectiveness of the methods in the structured design methodology has been enhanced over the years by additional techniques and automation support, but the basic concepts have remained unchanged. It is also interesting to note that these structured design techniques have been shown to work poorly in such areas as real time control and transaction based systems. One reason for this poor performance is probably the fact that the original set

¹Integrated Design Support System (see Section 1.5 of this report)

of systems which Constantine studied did not include any systems of this type. Thus, it should be remembered that any methodology (formal encoding of good practice) designed to support a development activity does not say anything about that activity in general. It only says that performing that activity within a particular context provides good results if a certain method is followed.

The existing IDEF theories and the SDM, which evolved from the ICAM experience, were also evolutions of previous methods. IDEF0 is a direct adaptation of SADT, which itself evolved from cell modeling techniques during the development of the AFCAM architecture in 1972. IDEF1 evolved from relational (Chen), network (Bachman) and hierarchical data modeling and database design concepts. IDEF2 evolved from the QGERT, SLAM, and OFD lineage of network based system dynamics modeling languages. The SDM established in the Air Force ICAM project 1701 was based on the ICAM experience, and the experience of the industry coalition in developing manufacturing, design and defense weapons systems. Unfortunately, the available experience base in 1981 did not include extensive integrated information systems development experience. Thus, while the basic concepts are solid, there are many specific integration issues or decisions that developers who are doing IDS developments, must address which are not supported in the current modeling or development methodologies.

Four years have passed since SDM ICAM Project 1701 was completed, and the IDEF techniques have been frozen for seven years. During this time an enormous experience base has been established in the use of these products to support the development of integrated information systems. Many contractors, and private industry users have developed extensions to the basic methods. In this report we have tried to identify, collect, and analyze, the experience of these system developers and their method extensions to provide the basis for our proposed plan. The plan reflects the fact that significant improvements over the existing methods can be achieved with little more than a formalization and method integration effort. During the last four years significant research results from the Artificial Intelligence community have emerged which offers the possibility of both improved techniques for research in methodology development and new software tools for building automated support for SDM and IDEF users.

1.3 Project Goals and Objectives

The following paragraphs summarize the objectives and goals of this effort. The goals were based on coalition experience with methodology development, experience of the DoD and aerospace community in the use of current system development methodologies and IDEF modeling methods, and the experience of the IDS implementation activities at Rockwell International.

Objective

The primary objective of this effort was to provide technical support for establishing a research plan for the development of "a structured approach for the strategic planning, tactical planning, requirements definition, design, construction, implementation, and maintenance" of large scale distributed, heterogeneous integrated information systems. For the remainder of this document the structured approach will be referred to as the Integrated Information System Evolution Environment (IISEE).

Goals of the "IISEE for IDS" Thrust

The driving force behind IISEE objectives and goals is the recognition, by the United States Air Force, and the coalition members of the Integrated Design Support System (IDS) development team, that for the IDS concept to be transitioned into widespread use an IISEE must be established which would allow system developers in the Air Force and its contractor community to evolve their current systems into an IDS framework. This evolution will usually involve: modifications to existing systems, replacement of existing systems, and/or the development of new systems. During the course of the IDS development, elements of the ICAM SDM methodologies, and particularly the IDEF modeling methods have been extensively applied. This application experience (as well as the experience of many other industry and government users) has uncovered voids in the original ICAM SDM, its component methods and in the availability of automated support for the development process. It is evident that the time is right for continuation of the SDM and IDEF methods development. It is also clear that such developments would be useful not only within the context of IDS goals but also for several other major DoD programs. Such a development represents a large scale effort, exceeding the capacity of any one program. Thus, staged funding must be acquired over several years from a variety of organizations.

The overall goal of IISEE thrust is to establish a structured systems development approach consisting of methodologies, methods, tools, techniques, and practices which can be used to:

1. Insure the success of integrated information systems development;
2. Increase the productivity of the project teams involved in systems development;
3. Improve the quality of the life cycle products of the development team;
4. Reduce cost and time required for integrated information system developments;

1.4 Organization of the Report

This report is organized into two logical parts. The first part summarizes the needs, requirements, state of the art voids and IISEE plan. The second part provides details of the technical research and assessment activities. The following describes the contents of

each chapter. The contents of the 10 appendices are described as referenced in the various chapters.

The remainder of Chapter 1 is focused on IDS information integration philosophy and the concept of an integrated information system evolution environment (IISEE). The IDS overview provides the technical context of the methodologies and tools we are considering. The IISEE overview provides an introduction to some of the key definitions and concepts which directed towards this work. The IISEE overview is meant to provide a high level picture of what the coalition perceives as the "end game" of the plan presented in Section 3.5 of this report.

Chapter 2 provides a summarization of key results of the analysis efforts in this project. The first section provides an overview of the problems which organizations face in attempts to achieve information integration of engineering and manufacturing information systems. This section also summarizes the problems with existing methods, and automated tools as perceived by systems engineers in various organizations. Finally this section presents a summary of organizational needs relative to methodologies, frameworks, and automated tools and environments for:

1. Strategic Information Integration Planning
2. Tactical Planning
3. Design
4. Construction
5. Implementation
6. Evolution / Maintenance

Section 2.2 summarizes the efforts of the project team to construct a model of the process of integrated information system evolution. The project team recognizes the need to produce an activity model to structure the process of identification of existing methods, automated tools, and to provide a basis for the structuring of the technology voids. Section 2.3 summarizes the results of the state-of-the-art assessment of methodologies and automated tools which are available to support the activities of the information system evolution models discussed in Section 2.2. Finally, Section 2.4 provides a description of the requirements for an IISEE based on company needs, the development process definition, and the state-of-the-art review results. Chapter 2 provides the basis for the plan proposed in Chapter 3 of this report.

Chapter 3 provides a plan for the development of the enabling technologies which will (with the existing methodologies and automated tools) provide an IISEE for organizations to implement and maintain information integrated engineering and manufacturing systems. Section 3.1 provides a justification for the the component and framework developments in the context of the IDS Program goals and objectives. Section 3.2 identifies the requirements

for framework and component methodology developments. Section 3.3 identifies the requirements for automated tools and environments to support the application of the framework and component methodologies. Section 3.4 identifies the requirements for technology transfer efforts needed to support the dissemination of the results of an IISEE program. Finally, Section 3.5 provides a series of project descriptions, with time and resource estimates for the development of the enabling technologies identified in Sections 3.2 through 3.4.

Chapters 4 and 5 present the results of some investigative research which was performed as a part of this effort. One of the major industry needs is for integration of the component methodologies within a unified framework. A major stumbling block to the accomplishment of this integration is the lack of a formal basis for any of the existing techniques. Chapter 4 presents the results of an effort to formally define the syntax and semantics of a number of existing systems analysis and engineering methods in widespread use today. The rationale behind this formalization effort is to:

1. Identify the informal intuitions that motivate the existing methods.
2. Provide a technical basis for the integration of existing methods.
3. Provide an objective basis for the comparison of methods to support usage planning.
4. Provide a technical basis and an engineering technique for the design of new methods.
5. Provide accurate specifications for the design of automated tools.

Central to the feasibility of establishing automated support for the integration, management and life cycle control of the data associated with a system development process is the capability to have a representation scheme for this information which is logically a superset of the individual methodologies. Section 4.4 describes the concepts, rationale, and preliminary requirements for such a representation scheme. This will be referred to as the "Neutral Information Representation Scheme" (NIRS).

Chapter 5 summarizes an investigation conducted into the use of natural language processing techniques for the acquisition of system description information, model validation support, and translation between methodology types. This section also describes exploratory work in the use of knowledge engineering languages, tools, and environments as the basis for construction of the IISEE automated tools, and as a basis for improvements to the fundamental IDS technology itself.

The ten appendices contain detailed information resulting from the analysis activities associated with this project.

1.5 IDS Technology Concepts and Implications

The Integrated Design Support System Program is focused on the capture, management, and communication of product technical data from its initial creation in design, through

realization in manufacture, to logistics operations (see Figure 1.1). The extent of the challenge facing the IDS in the defense system arena spans hundreds of contractors, world wide development and operation, war and peace situations, and a product life cycle which can easily span twenty years. Obviously the information technology across this extent of organization, space, situation, and time will vary significantly. One of the major targets of IDS is to isolate the product data from the changes in the information processing technology so that as new technology emerges it can be quickly and reliably inserted into the process (see Figure 1.2). The major data control issues which the IDS Program addresses are listed in Figure 1.3. The immensity of this task is somewhat mitigated by the fact that only critical product data needs to be controlled (see Figure 1.4).

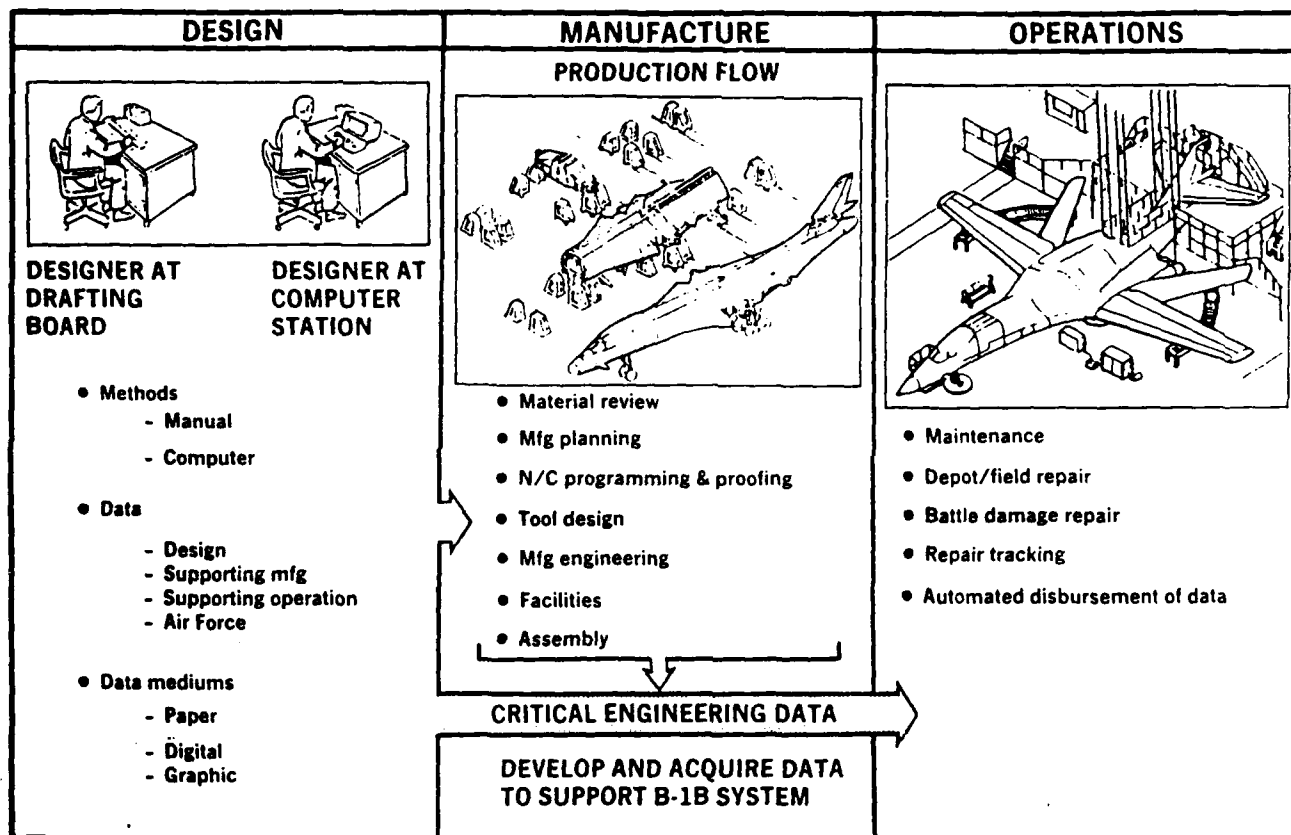
The approach taken in the IDS development is an implementation of the ISO/ANSI three schema architecture concept utilizing commercially available software. Under this approach, enterprise information control is accomplished via the establishment of a "conceptual schema" and an "executive control system." The conceptual schema contains a definition of the enterprise information management and control rules (see Figure 1.5). The logic behind the IDS concept is that the executive control system validates each transaction against these rules. This removes (in concept) constraint checking from the AP's. It also centralizes and provides the basis for the control of critical data in the organization.

However, this requires definition of an enterprise conceptual schema.² It also requires changing the way applications are designed and built. One of the primary changes required is the enforcement of more uniformity and control on the system development methodologies used within the organization. However, before such uniformity can be demanded a workable set of component methodologies and development frameworks must be engineered. This is one of the driving factors behind this effort. The IDS has been prototyped and demonstrated. From this experience and the experience of other programs (e.g. the AFWAL/MLTC IISS) it is known that there are key methods missing. It is also known that the complete set of methods needed would be too expensive to implement in a manual fashion. An automated set of component tools is needed to support each of the methods, and an integration framework to allow movement of data between tools is also required. Finally, a life cycle data control mechanism is needed to manage the evolution of that process. These needs and others will be described in later sections of this report.

Prior to attempting any description of the IISSE concepts, it is important to understand the context within which it is anticipated that an IDS shell will likely be implemented. While there are similarities, there appear to be some major differences between the planning, design, development, implementation, and maintenance of systems for managing and controlling engineering/manufacturing technical data (at which the IDS is primarily directed)

²An enterprise conceptual schema can be roughly equated to a definition of the common data in a company which is used to govern the access, update, and use of that critical data. Such a definition is complex and costly to develop using current methods and tools. However the potential payback in terms of reduced product cycle time and improved organizational responsiveness has been shown to far outshadow the costs

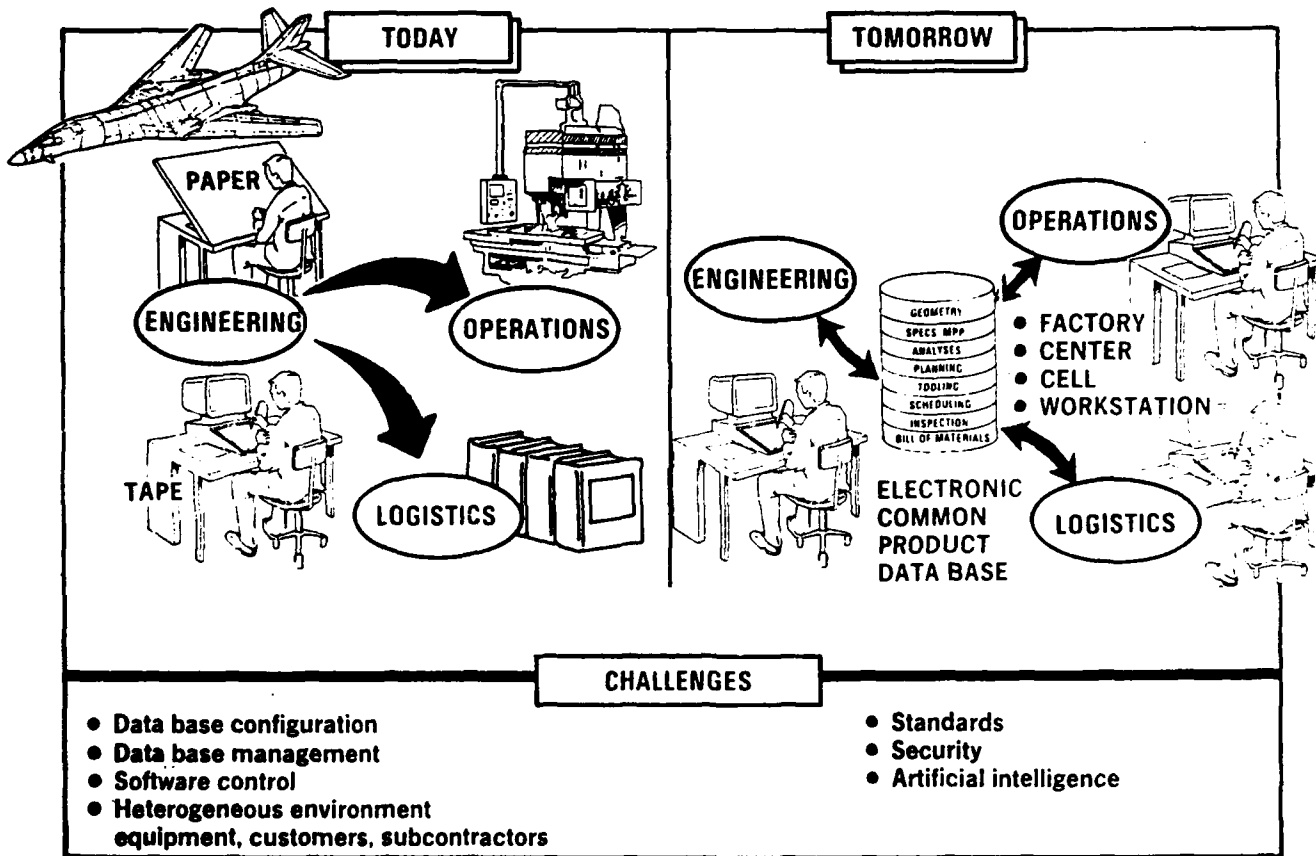
B-1B/IDS PROGRAM APPROACH



DS1-84-0450

Figure 1.1: IDS Program Approach

TECHNICAL INFORMATION RESOURCE MANAGEMENT



DS-85-6005A

Figure 1.2: Technology Insertion Under IDS

IDS ISSUES

- Critical technical data for life cycle support
- Configuration management and control
- Data structure
 - Logical
 - Physical
- Security
- Data base access
- Requirements for standards
 - Data structure
 - Data communications
- Life cycle cost

DS-85-2143

Figure 1.3: Data Control Issues

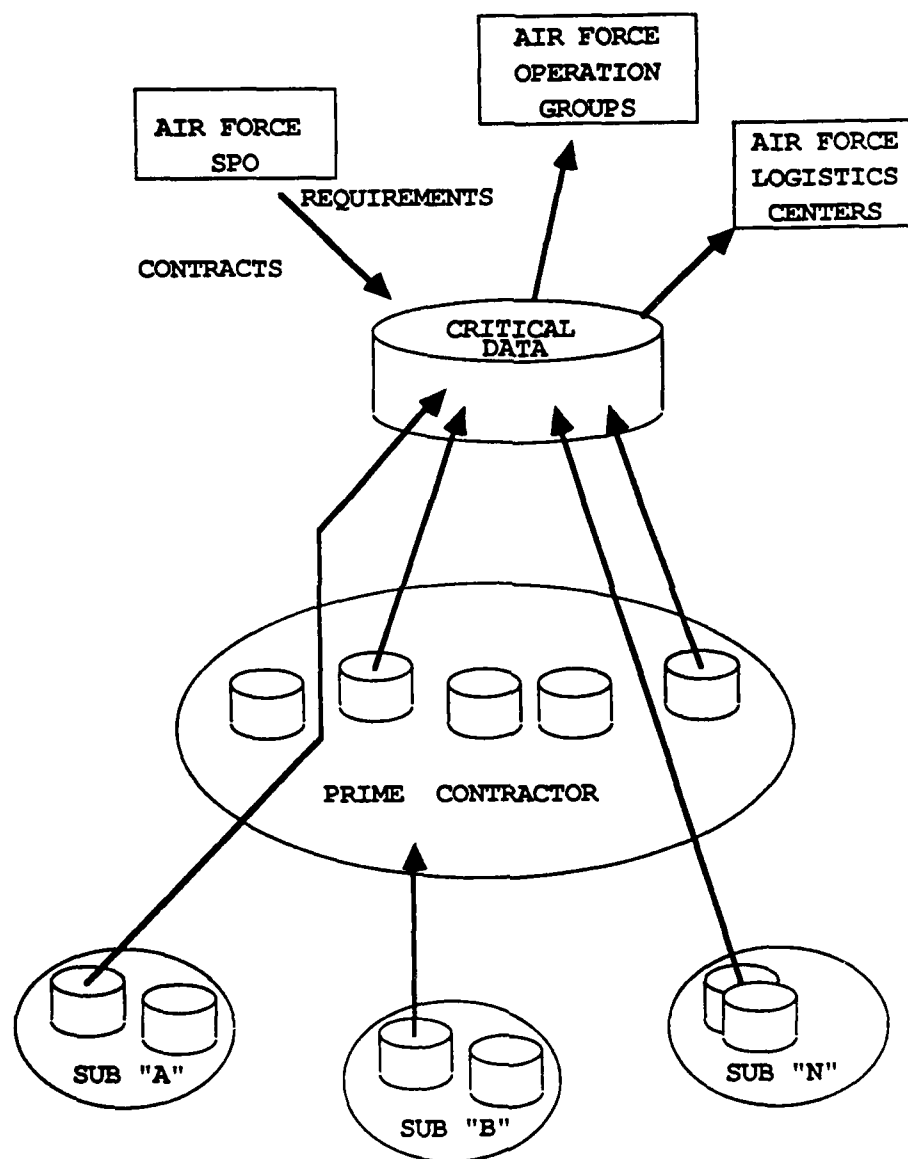
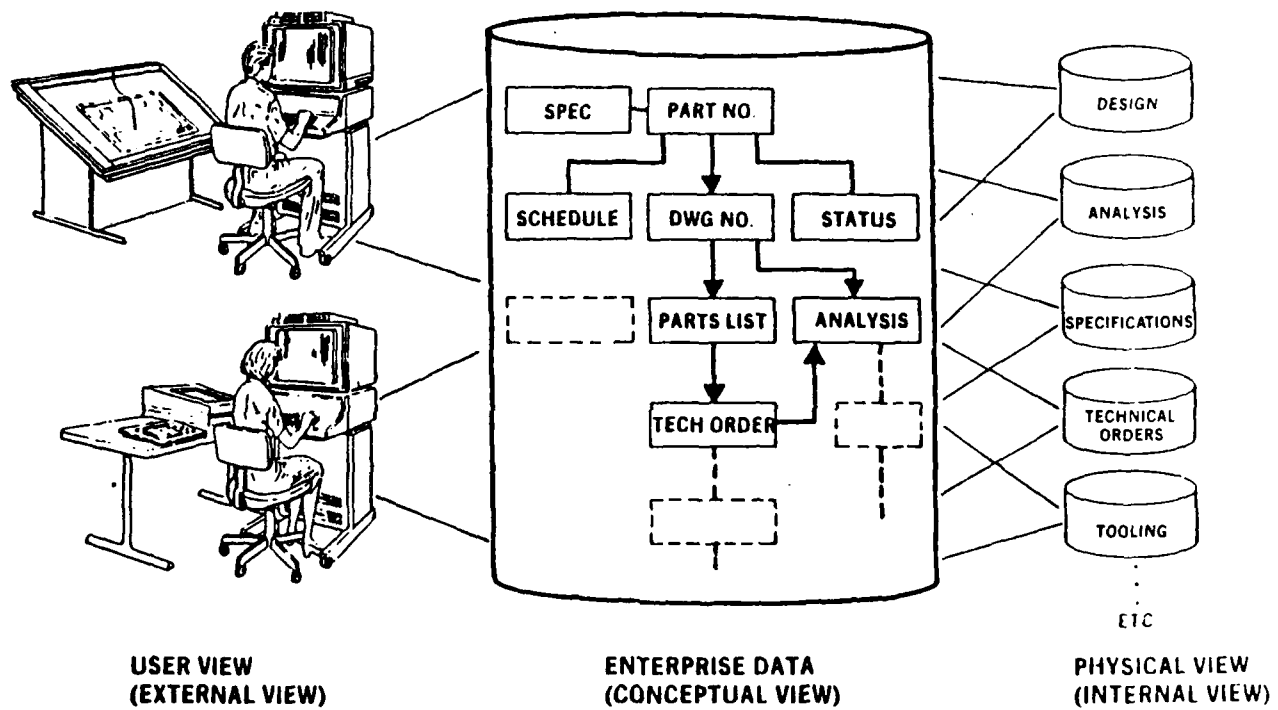


Figure 1.4: Control of Critical Data

IDS DATA BASE ARCHITECTURE



DS1-86-0175

Figure 1.5: IDS Three Schema Architecture Concept

and some other forms of business oriented data. These differences are associated with the complexity of the data, user interfaces, and techniques/tools that are used to evolve the data. Technical data has a distinct life cycle that evolves over time with many inter-dependencies that must be tracked and controlled and which correlate to an emerging product definition. For example, technical data starts as a definition of an initial concept that identifies the target product's basic mission, such as performance, cost and weight requirements. Preliminary design data describes the basic product structure which can then be simulated to determine how well the preliminary design can achieve the target requirements. Project Management is concerned with the evolution status of the product design at key decision points throughout its entire life cycle. Project Management approves each step in the product design/development process. When new designs are completed, changed, and/or released, appropriate organizational groups must be notified. Since there is so much technical data associated with complex mechanical products such as aircraft, space stations, ships, etc., it is essential to develop systems that are capable of managing and controlling the evolution and maintenance of this type of data. The following paragraphs further describe some of these complexities:

- Data Life Cycles - technical data evolves through life cycle states. Different technical data types are dependent upon each other and must satisfy certain constraints before their state changes are approved.
- Configuration Management - products have a specific structure (configuration) that must be managed throughout the product life cycle. Technical Data describes the various components that comprise the product structure. Changes to the product structure must be managed to insure that the proper component designs are released for production.
- Status Accounting - since technical data is critical to the evolution of the product, its evolution status (state) is important to the project/technical managers and engineers expecting to use the data as it evolves.
- Complex Applications - technical data (geometrics, simulation, test) is produced by applications that are graphically oriented; i.e., they input/output 2 dimension, 3 dimension, solid representations, scatter grams, line curve charts, etc. These applications typically maintain their own data structures and normally do not share data because it is in different formats.

Hardware/Software technology is continuing to evolve. Powerful single user engineering workstations are becoming economical to obtain and use. This trend establishes a need to deal with distributed data that is being developed by heterogeneous applications that may or may not be communicating with a central system. Thus, the IDS system development process must also deal with the distributed nature of the data as well as its complexities, life cycle control characteristics, and relationship to other kinds of business data.

These are just a few of the complex issues that must be accommodated by an IDS shell. A facility which adequately addresses these issues is likely to address the data management issues associated with other, less complex forms of business data as well.

1.6 Terminology

The objective of this section is to provide a definition of the terminology which will be used in the remainder of this report, and to provide a high level view of the IISSE concept. A more detailed description of the IISSE concept is provided in Section 2.4 of this report.

As with any emerging discipline, systems engineering is still in the process of resolving the terminology associated with the concepts inherent in its theory and practice. This lack of terminology standardization is a major stumbling block for advancement of the state of the art of this discipline. For the purpose of communication among the coalition members, we adopt the following definitions for the concepts presented in Figure 1.6.

1. Framework – A collection of *methodologies* tied together by a *development procedure* to support the development of a particular type of system in a specific type of organizational environment. A framework provides the overall integration method for an application in an environment. Because of this role it will be referred to as an **Integrating Framework** in the plan section of this report.
2. Methodology – A collection of *methods* with similar semantics and application areas (e.g. an *Information Modeling Methodology* would include ENALIM, IDEF1, EER, IDEF1X, etc.).
3. Development Procedure – A collection of decisions and actions organized into steps and phases which describe the activities required to develop or evolve a particular type of system. The development procedure would call for the use of many methods as well as define the decisions to be made with the results of application of those methods. Where possible the development procedure will call for the use of a *methodology* indicating that any *method* within that class with certain characteristics would be acceptable.
4. Method – A defined set of concepts, along with a discipline, which would guide the application of concepts for a particular use in support of a particular system development life cycle activity (e.g. the IDEF1 method or the IDEF0 method).
5. Definition – The collection of *concepts* and the *theory* of how the concepts work together to map onto a particular aspect of reality. The *definition* component of a method essentially consists of both the formal and informal description of the method.
6. Concepts – The motivating intuitions or basic modeling elements of a method (e.g. entity classes, relation classes, and attribute classes in IDEF1).

7. Theory – The logic which describes how the basic concepts of a method work together to represent a particular view of reality (e.g. activity, information, process etc.) to support a specific system development activity (e.g. planning, analysis, design etc.).
8. Discipline – The description of how to acquire information about a system and organize/display that information using a particular method. The discipline component of a method contains the practical application instructions (i.e. *How to do it*) for the method.
9. Syntax – The description of the symbols and presentation rules of a method (e.g. boxes and arrows and decomposition rules in IDEF0 function modeling).
10. Procedures – The description of the activities which a modeler must go through in order to collect, organize, analyze and validate information about a system in order to build a model using a particular method. The procedure component normally consists of procedure descriptions, data collection / analysis forms, and library organization guidelines.
11. Use – A description of the activities which a system developer performs to analyze a completed model and extract information necessary to perform the next step in the system development process.
12. Environment – Refers to an automated system provided for the support and integration of the overall life cycle activities associated with the planning, definition, engineering, design, implementation, and evolution of an information system. The environment is considered to include the data management utilities, interface utilities and control mechanisms required to support the use of the tools and computer languages in an integrated manner. The environment can be thought of as the automated version of an integration framework for the support of the system development team.
13. Tool – In the context of this report, this refers to any automated mechanism or implementation of a modeling or programming concept which is too complex or specialized to be considered as a part of a base language.
14. Computer Language – In the context of this report, this refers to the procedural, functional, logic, object oriented, or descriptive languages which are used to construct an IDS like information integration system or the applications within such an environment. The construction and use of an IDS based system requires many special purpose languages (e.g. schema definition languages, user interface definition languages etc.) as well as traditional programming languages (e.g. COBOL, FORTRAN, C, LISP etc.) all of which must be supported by an automated system development support environment.

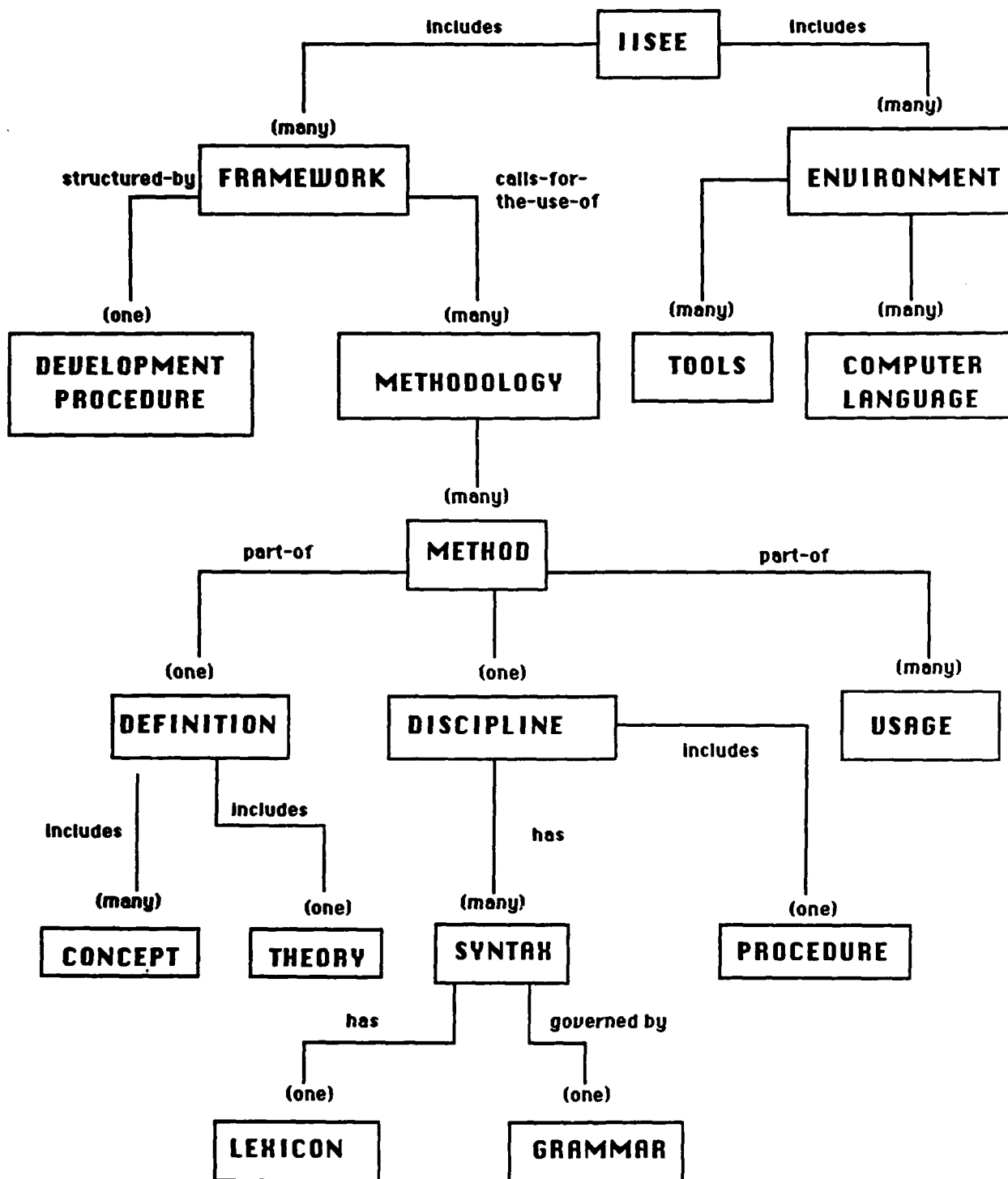


Figure 1.6: Key Terminology Relationships

1.6.1 IISEE Overview

If we invoke a product view of IISEE and look towards the future (after the plan outlined in Section 3.5 has been put into place), we can imagine a systems developer in an organization placing an order for an IISEE. What might arrive would be a diskette which would contain a small expert system which would help him to relate his current situation to the "standard" frameworks provided by the IISEE product line. Once the user has adequately responded to the questions of the program, a configuration of product and software would be produced which could be ordered from DTIC. Presuming the user places such an order, he would eventually receive a large carton whose contents are illustrated in Figure 1.7.

The "Guidebook" item in the carton would contain an overview of the philosophy, assumptions, and structure of the approach to systems development embodied in the IISEE. This document also contains a description of a method for analyzing specific system development needs so that he can accurately use the "Guide" expert system. The "Guide" would lead the recipient through a series of questions which would result in a recommended selection of one of the smaller boxes in the carton. The smaller boxes each contain a framework, complete with a development procedure and a collection of methodologies. (*Obviously in some cases the specific method referred to by a step in the development procedure may not be included since it is a commercial product.*)

In the bottom of the carton is included a set of tapes and another diskette. These tapes contain the IISEE system/software development factory. That is, the system development environment (SDE), component tools, and languages required to provide automated support for the execution of a framework. The setup diskette contains another expert system which identifies what hardware, commercial software, and people skills are required to setup and operate the factory. It also contains the instructions for configuring the factory for the particular framework chosen.

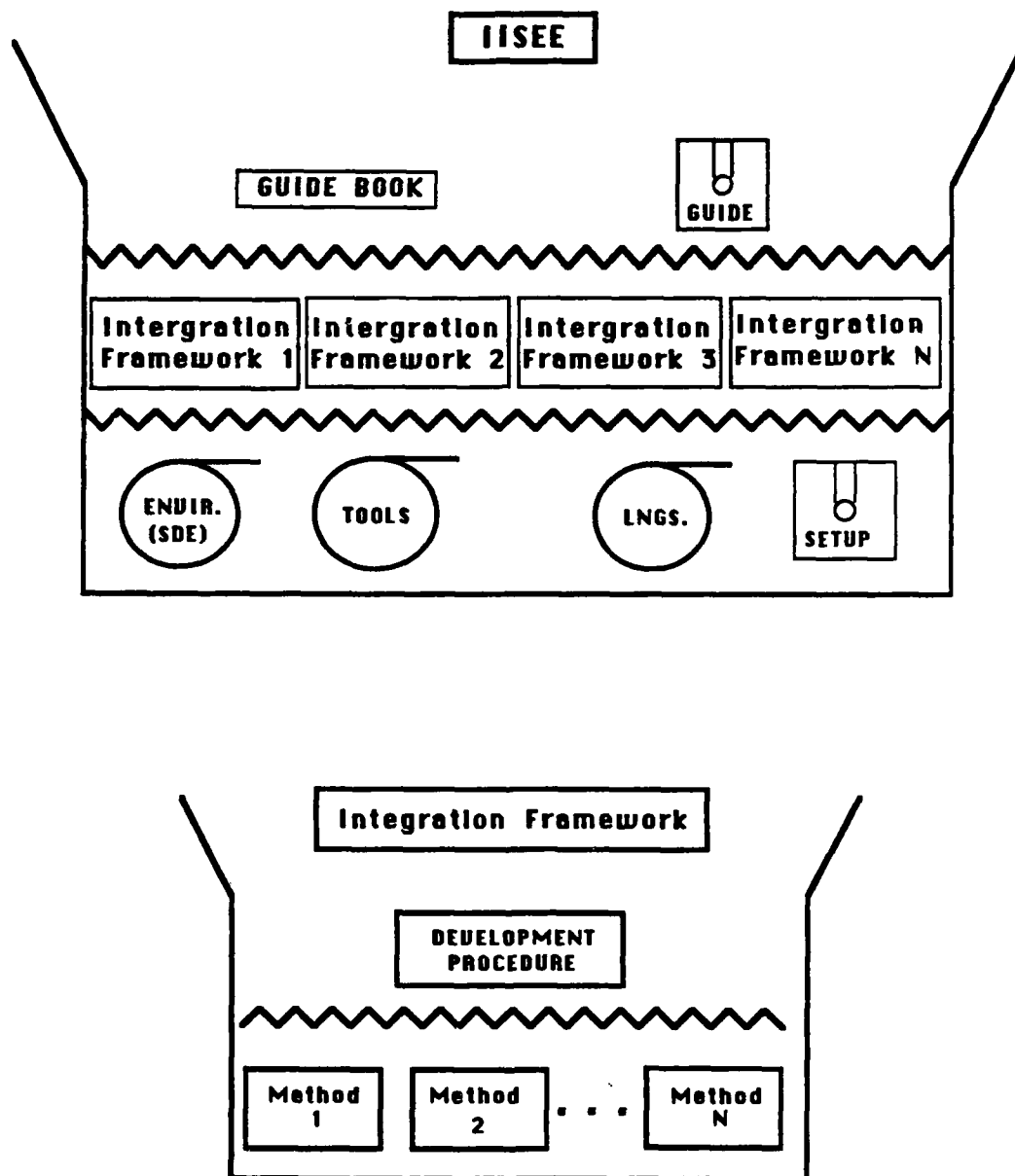


Figure 1.7: An Instance of an IISSE and an Integrating Framework

Chapter 2

Information Engineering Needs, Process, and SOA

When most of us think of an engineering/manufacturing project, three issues come to mind: planning, management, and control. Planning is the process of preparing for the commitment of project resources in the most effective manner. Management is the process through which project goals and/or objectives are achieved. Planning involves the coordination of group activities, wherein management requires plans, organizations, staffing, direction and imposition of controls to achieve an objective with constraints placed on time, cost, and performance. Control is the process of making events conform to a schedule by coordinating the actions of all organizations according to the plan established for achieving the objective.

For engineering and manufacturing to operate cost effectively, there must be management and control of technical product data. This has become an increasingly critical aspect of Computer Aided Design/Computer Aided Manufacturing (CAD/CAM) activities within engineering and manufacturing environments. When engineering and manufacturing processes are conducted, a variety of data management and control problems may be encountered (i.e., the problems that occur when controlling engineering changes, accessing data, tracking data, and managing the product configuration, etc.). Currently, systems that manage and control technical product data are non-integrated, thus causing engineering and manufacturing project planning, management and control as well as CAD/CAM to be decentralized, time consuming, and costly processes. The IDS technology provides a mechanism for achieving the needed integration of this technical data. The primary issue before companies today is how to implement integration programs which can take advantage of this technology.

2.1 Existing Methods and Needs

This section addresses issues, problems and voids associated with the use of existing (application engineering based) methods for developing and implementing integrated information

systems. It also discusses generic company needs for integrating frameworks, component methods, automated tools, and technology transfer mechanisms.

2.1.1 Issues

Very little experience is available when it comes to conducting the level of analysis that is required to establish an overall enterprise strategic or tactical plan for the development and implementation of integrated information systems within engineering/manufacturing environments. The function, information and dynamics modeling methodologies (i.e., IDEF0,1,2) being utilized today were designed to be used as techniques that document requirements for establishing integrated information systems. These methods have been used on several Air Force and commercial programs with varying degrees of success.

Major corporations that conduct business with the DoD in the future will be required to develop integrated information systems that can communicate over distributed networks. In order to accomplish this, these companies will have to develop strategic, tactical and operational plans for implementing these types of systems. Once the plans are approved, they will be initiated as programs with many tightly related projects to either develop or obtain the required information or system integration technology, and then to implement applications within it. This is a major undertaking that requires a comprehensive integrated information system evolution environment that can be used throughout the design, development and implementation phases of the project.

Several issues relating to the "AS IS" information environment in engineering / manufacturing industries have been identified that pertain to the design, development and implementation of integrated information systems. Examination of these issues helps to expose the need for effective methods that support each phase of the system evolution life cycle. These issues are as follows:

- Paper Volume - most major engineering/manufacturing corporations maintain large volumes of technical data (e.g., drawings, specifications, technical reports, etc.) that exist on paper. Many of these companies are in the process of converting these data to electronic form using a variety of non-integrated applications that operate on computer systems such as PCs (personal computers), mini computers and large host computers. There is growing concern about the global management and control of this data.
- Non-Integrated Systems - many systems exist within engineering / manufacturing companies that were developed independent of one another. In most cases, these systems utilize different operating systems and data base management systems. This means that they do not share the data that they utilize and more often than not will duplicate the data causing consistency problems. Existing systems must be factored into any integrated information system strategy.

- **Cultural Impact** - for the most part, company employees have not been mentally prepared for the technological evolution of integrated information systems. In most cases, corporate management and individual department managers within the enterprise do not understand the importance of information or system integration. Even in cases where integrated information systems are beginning to be implemented, end users are reluctant to support them. This reluctance can be traced to at least three sources:
 1. Apparent loss of control by the individual organization of common data.
 2. The additional time and effort of development of integrated applications.
 3. The loss of local flexibility to change policies and rules governing the data.
 4. Potential degradation of response in individual applications due to access of common data.
- **Evolving Technology** - in recent years computer technology has proliferated. Much of the application functionality that was once located on host systems is now being employed at the workstation level. Similarly, data that was once stored on mechanical disk devices is now being archived on large volume optical storage devices. In addition the emergence of knowledge based systems in manufacturing and engineering applications is introducing a new type of common data which must be managed. These phenomena further complicate the information integration issue. Methods need to be developed that can effectively model distributed data and knowledge as well as the network structures it will be transmitted over.
- **Knowledge Attrition** - in recent years there has been a gradual disappearance of engineering/manufacturing knowledge. Engineering and manufacturing expertise that has evolved over the last thirty years is beginning to erode within many companies due to the retirement of those individuals that have many years of experience in design and production processes. In the past, a design engineer may have derived his experience from earlier manufacturing experience. This trend towards erosion, in conjunction with rapid technological advances, is causing major problems within industry. Once knowledge disappears, it is very difficult to regain it through training and education. Engineering and manufacturing expertise needs to be captured in a corporate knowledge base that can evolve over time and be accessible to a new breed of engineers that are emerging from our educational institutions.
- **Global Product Data Knowledge** - since information systems are not currently integrated, it is difficult if not impossible to obtain the status of evolving technical product data. Management must usually wait long periods of time to obtain status information pertaining to critical technical data. This is due to the lack of a global configuration management and control capability that is capable of tracking and reporting the life cycle states of evolving technical data related to the product.

- **Data Longevity** - the life expectancy of military weapons systems spans 20 years or more. These weapons systems are subject to change or modification rather than new design and development. The technical data that is associated with these weapons systems must remain accessible throughout the entire weapons system life span so as to facilitate quick and cost effective design modifications or repairs that may be required.

2.1.2 Integrated Information Management and Control Definition Needs

When determining integrated information system requirements it is important to realize that each enterprise is different and each has its own set of management policies and operational procedures. Changing these policies and procedures in order to satisfy the requirements of a "general purpose" system would be a disastrous step. Also as the needs for an IDS like information integration mechanism become more wide spread the number of "vendors" who want to fill those needs will increase. Without a standard definition of what functionality must be delivered by such an information integration mechanism company management will be unable to make informed decisions between suppliers. What is needed is an integrated information system definition (a standard IDS description) that can be customized to support data management and control needs of the enterprise. A set of typical concerns that a system such as this would have to address are as follows:

- **Engineering Change Control** - Authorized engineering changes made during product design processes are costly in terms of the time and money that is expended to actually make the change. For instance, changing a part design that is stored in one file may directly affect many related part designs that are stored in other files. Each time a part design is changed, the changed part design and the related part designs that are affected by the change, must be reviewed and approved by design, analysis and test engineers assigned to the project management team. It is essential that part designs changed and/or affected by a change be properly controlled so that updates, reviews and approvals can take place in a logical and efficient manner. Notification and approval messages must be generated when changes have occurred to the product data.
- **Data Access** - Early in the design process, engineers should be able to easily create, access, and change part design files. In the later stages of the design process, part design files should be protected from unauthorized access. This means that engineers working on the project should not be able to access and change the part design files that have been reviewed, approved and released for analysis, prototype and test activities. If unauthorized design changes occur due to a lack of access control, the time, money and effort expended to rectify the problems that are caused by design errors can be enormous.

- **Data Tracking** - Part design data must be tracked so that it can quickly be located and retrieved for evaluation when design problems are uncovered during analysis and test activities. On a large project, this becomes a major problem because part design data proliferates when the product structure is complex, i.e., composed of many parts. To compound the problem, the analysis and test runs create additional volumes of data that also must be tracked. When design, analysis and test data can not be located, major project delays occur and design errors go undetected.
- **Product Configuration Management** - a product is made up of many parts, which in themselves may be made up of many component parts and/or assemblies. The relationship that exists between a part and its component parts is called the product configuration (structure). It has become increasingly important to maintain the integrity of the product configuration throughout its life cycle. Being able to identify which component parts are associated with a particular sub-assembly, which sub-assemblies make up an assembly, and what life cycle design process state of the each component part exhibits, provides the project management team with the necessary information to effectively evaluate the evolving product design.

A set of more specific requirements of an integrated information system definition are as follows:

- **Manage data at the file and/or database level.** The system must be able to manage various types of product data objects: CAE geometry, CAD drawings, analytical models, test data, specifications, etc.
- **Manage data from multiple computing environments.** Most engineering organizations utilize more than one computer to perform various functions. The system must be able to locate a data object within the network of computers and make it available to the user.
- **Manage data from multiple organizations.** The system must track the location of distributed data that is outside its own environment. Furthermore, it must be able to notify authorized users when data has changed and/or is affected by a change.
- **Enable the definition of multiple relationships between data objects.** Engineering data objects (in this case parts) are interrelated and form a product structure. Geometry, analysis, and test data are related to parts within the product structure. A change to any one of the data objects may affect the others; thus, it is important that the system maintains relationships between data objects.
- **Control data object access.** The system should limit user access to product data objects. For example, a design engineer should have "read/write" access to his own product data files when they are in the "pre-release" design phase. This same design

engineer should have "read only" access to another design engineers product data files. Once the product data files are in a "released" state, users should only be able to obtain copies of the released files.

- Control data release. The system should track the life cycle status of all product data files. In order to accomplish this, each file must be assigned a set of life cycle states through which it will evolve. The states should be defined by the project management team according to existing policies and operating procedures.
- Provide query commands across the network. The system should provide the user with the capability to query any data that exists within the system. For example, one might ask for a list of all the CAD drawings that are in the system, or all the analysis runs that were performed on a particular part geometry.
- Generate messages. The system should generate both informative and action messages that alert project team members when the status of a product data file has changed or when review/approval is required. The system should allow project team members to interactively respond to the messages.
- Control formats. The system should enable project team members to design menus, data input screens, and printed report forms which are similar to those forms already in use. By emulating existing paper forms, reports and procedures, the time it takes to teach system functionality can be reduced and political barriers can be avoided.

2.1.3 Methodology Scope Expansion Needs

The ICAM System Development Methodology Project (December 1985) did not address the *Administer Business Information Strategy*, *Administer Program* or *Manage Project* activities. These activities are an important aspect of the design, development and implementation of integrated information systems. They represent the strategic and tactical decision making process. Companies need a methodology when entering a new business venture that will help assess the affect on the entire enterprise information strategy. The company should develop an information strategy consistent with thier business strategy by first evaluating the strategic concepts, followed by a definition and prioritization of strategic objectives. The information strategy should be developed by first planning, then authorizing, then controlling strategic projects. A strategic project may be the design, development, and implementation of an integrated information system. If so, companies need a methodology of planning and managing the evolution of such a system. This is a complex undertaking that involves the development of an evolution strategy. The evolution strategy should take into account the impact that a system such as this will have on the business strategy as well as the business operations. The evolution strategy should also include a comprehensive tactical plan that addresses development, management, service, and resource planning.

2.1.4 Framework Needs

Engineering/manufacturing companies generally tend to have their own unique way of conducting strategic, tactical and operational planning for the implementation of information systems. Now that these companies are becoming more aware of the importance of integrated information system technology, there is a vital need for a cohesive framework that guides management through the various levels of strategic, tactical and operational planning that is required for effective analysis, design, development, and implementation of integrated information systems. Integrated information systems technology is a new technology that involves the entire enterprise including upper level management, engineering, and manufacturing. Traditional methodologies are not robust enough and are primarily oriented towards stand alone application software development projects rather than enterprise wide information system integration. In short, the framework required to conduct the depth and breadth of effort required for an information integration program within a company are not currently in place.

Since there are few, if any, methodologies available for conducting enterprise wide strategic, tactical and operational planning for integrated information systems, companies are looking to outside consulting firms for support in this area. In house information management services groups are not trained in the use of structured methodological techniques to adequately address the strategic, tactical and operational planning needs required to develop and implement an integrated information system. High level management does not understand information integration technology and, in many cases, the importance of information as a corporate resource. The current trend is to purchase hardware and software as an information management solution. The only problem is that the hardware and software cannot easily be integrated with existing systems. Getting started in the right direction appears to be the biggest hurdle facing the company that wishes to address its own information management problems. Strategic, tactical, and operational planners know that they want to reduce costs, shorten product planning, design and development cycles, and improve their overall ability to maintain the product over long periods of time. Their ability to translate these goals into feasible plans often fails because the individual methods do not guide them in the proper overall direction. The problems that have been encountered by planners tend not to be with the effectiveness of the methodologies, but that the lack a structured approach to the strategic, tactical and operational planning for the design, development and implementation of integrated information systems.

In Appendix B of this report, the reader will find a node tree entitled "Integrated Information System Evolution Process." The node tree outlines the strategic, tactical, and operational planning activities that are necessary to effectively design, develop, and implement an integrated information system. Following the node tree is a set of function (activity) models that graphically depict the process. Included with the node tree and function models the reader will find a list of activity descriptions that describe what happens during each function depicted on the models. The overall intent of the node trees and models is to the

illustrate the breath of activities which must be addressed by a structured approach.

2.1.5 Common Method Needs

Application of existing methods are often site specific. In the following paragraphs, only the general problems that apply to all applications will be discussed. The specific method voids will be addressed in the following section. The most obvious method needs / problems that have surfaced during this analysis are as follows:

- The methodologies do not have user levels associated with them. For managers to understand and implement a method, they must subscribe to and acquire the same level of training as the analyst who will be responsible for in-depth application and analysis work. What is needed is alternate syntax for different levels of users. For example, the process of application of the IDEF1 method is effective for detecting information which should be acquired and managed by an organization but is not currently. Managers need to know which of their current problems are caused by these information gaps. The current method does not have a *use* procedure which provides a suggested syntax for display of such an extraction from a completed model.
- The integration of methods has never been accomplished. This is primarily because of a lack of establishment of an engineering formalization of the syntax and semantics of the basic concepts and theory behind the techniques. Due to this integration issue, it is difficult and sometimes even impossible to directly utilize data from one modeling method or procedure directly to another. For example, this is especially true of data transfer from IDEF0 to IDEF1.
- The current methodologies are difficult to describe beyond the level of mechanisms on an IDEF0 modeling methodology. The underlying principles and theory upon which the modeling methods are based are not easily recognized at the application level.
- Current methods are very inflexible and do not provide the user with any options during the application phases of the methodologies. A user must conform to the outlined procedures prescribed in the methodology or the results may lose their meaning.
- Many of the existing methods are deterministic and do not provide easy application when the system, functions, and/or information being modeled are stochastic (or in any way time dependent) in nature.
- Current methods are not structured to allow for the dynamic properties of information. Some provision is needed for identifying the age of data at the input side of the system, and to provide for the aging process within the structure of the method.

- Most of the current methods have not been developed with the objective of knowledge base implementation. Future methods must be developed with the objective of generating and populating a knowledge bases. It is critical for methods to be made compatible and/or be integrated in order to accomplish this.
- Current methods have not been developed for a single set of application based data standards. They all presume complete autonomy of the individual application developer to define his own private standards.
- At the application level, the concept of data acquisition for the population of a model or use in a methodology is not clearly linked to the data administration process nor is there any relationship to the system development process. As a result, when the design is implemented, the data is not neatly available in readily accessible files.
- At the discipline level, no existing methods have the automated support needed to improve the actual model development decision making process. Existing tools mainly provide automated features that store a finished model and/or draw graphics representations. However the key processes of classification, consistency and completeness analysis, common data identification, validation review and model integration are largely ignored. This is a common problem and results in both long application lead times for stand alone applications and prohibitively expensive integrated systems developments.
- Enterprise analysis procedures cannot take advantage of the data produced by current methods for applications engineering. Only after some form of data transformation and/or data analysis can that data be put in a form useable for enterprise decisions.

2.1.6 Component Method Voids

The following paragraphs describe specific component method voids which were uncovered in this needs analysis effort.

Systems analysts and consultants who have been working on the analysis, design, development, and implementation of integrated information systems (e.g., the Air Force Integrated Design Support System (IDS)), have had difficulties mapping information between the strategic, tactical and operational models that were developed during each of the respective activities. This difficulty stems from the fact that the methods that are available to them are not designed as an integrated set nor were they designed to support an information engineering paradigm for system development. Also, the methods fail to give clear guidelines on how to apply the associated concepts in the areas of strategic, tactical and operational planning for the implementation of integrated information system technology. Most of the guidelines that are available apply to usage as requirements specification methods, and not as planning, design, or implementation methods. In many cases this is justified because the concepts embodied in the methods were never intended to be applied in those domains. But

even in these cases the repeated requests by the practicing system developers indicates a void in the total set of methods required.

The function, information and dynamic modeling methods in use today are very generic. They can be used for a variety of purposes. There are few guidelines on how to effectively collect the data object types that are used to populate an integrated information system knowledge base. There is a need to develop standards for the types of data objects that are required by such a knowledge base. Most of the methodologies available today collect a limited set of data object types such as entity definitions, activity descriptions, etc. There is a need for a much broader set of data objects that includes the ones that follow:

- Data Objects
- Time Dependent Data Object Flows
- Time Dependent Process Flows
- Data Object Constraint Rules
- Organizations
- Organization Structures
- Project Codes
- Legal Value Sets
- Data Object Approval Rules
- Notification/Distribution Recipients
- Data Object State Transition Rules
- Data Object Access Rules
- Data Object Revision Reason Rules
- Task Initiation Rules
- Menu Access Rules
- File Location Rules

No methodologies are available in industry today that can effectively collect, analyze, define, verify, store, maintain and display the semantics (i.e., the meaning of model elements) associated with product data and their life cycles. Many of the system engineering tools that are available today, process model element input in a syntactical fashion, i.e., they do not

interpret the meaning of model elements. There is a need for tools that have the ability to interpret the meaning of data, thus, processing input semantically as well as syntactically.

To clarify the point, an information model contains syntax such as boxes, lines and circles, which semantically represent entities, relationships between entities, and entity relationship cardinality. It is these semantical representations that a semantic data engineering tool should be able to interpret.

There is a need to capture information model relationship types (e.g., a-part-of, an-instance-of, a-kind-of, connected-to, is-used-for, sends-to, receives-from, etc.) that support the kinds of engineering and manufacturing applications that deal with the problems related to part geometry, product structures, machine fault diagnosis, scheduling, process planning, etc.

Of equal importance is the need to capture the semantics that are represented on function models, i.e., the activities that must be performed in a given sequence, the life cycle states that as the product data evolves, the conditions that must be satisfied and the mechanisms that are required to perform the activities must be known. This information is required so that product data can effectively be managed and controlled during its life cycle.

2.1.7 Tool And Tool Environment Problems

Several Computer Aided Software Engineering (CASE) computer based tools have appeared during the last two years. Some of the tools are claimed to support the strategic, tactical, and operational levels of system engineering. Many of the tools are developed on top of a data dictionary that provides some level of consistency checking between the modeling capabilities that are provided by the tool. The tools usually provide two dimensional graphics capabilities for producing data flow, entity relationship, and system architecture models. Other tools provide screen layout and in some cases data base generation and prototyping capabilities. These tools are being used by many companies to provide requirements specification and structured design documentation for application engineering approaches to information systems. Section 2.3 contains a summary of the state of the art in CASE tools relevant to the information engineering approach to information integrated systems development.

Little or no documentation/training is provided on how to apply these tools for performing strategic, tactical, and operational planning for integrated information system technology or for carrying out design, development, and implementation in an information engineering fashion. Furthermore, no data format standards exist for sharing data / information between tools developed by different vendors. There is a critical need to identify the different types of data object types that are required by an integrated information system knowledge base. The following paragraphs describe in more detail the tool capabilities that are required to support the evolution and implementation of information integrated systems:

- State Transition Rule Consistency Analyzer - automated support for capture, representation and analysis of data object state transitions over time with the ability to

indicate precision (months, days, hours, minutes, etc.). This includes interdependency rule specification, e.g., data object "A" cannot change state unless data object "B" is in state "X" and data object "C" is in state "Y".

- **Data Object Scheduling Simulator** - the ability to determine the effect of data object scheduling requirements (both qualitatively and quantitatively), i.e., an object must be initiated by a given date and completed by a given date.
- **Performance Simulator** - the ability to assess the performance attributes associated with a data object, e.g., speed, dimension, weight, etc. This capability is needed for representing the characteristics of a mechanism data object type where a mechanism might be a program, machine, robot, etc.
- **Configuration Management** - the ability to support model versions, design alternatives, audit trails, change notifications, approvals, version control, access authorizations, status accounting, etc.
- **System Simulation of Architecture** - the ability to provide intra and inter component structural consistency and performance analysis. For example, a control model simulation would highlight bottlenecks associated with the inability of a particular subsystem configuration to support the required data state transitions.
- **Requirements Traceability** - the ability to trace strategic level requirements through tactical project requirements to operation specifications.
- **Prototyping Tools** - the ability to produce rapid prototypes from the models contained in the data dictionary to validate requirements. For example, code generation from structure models, menus/screens from user interface models, data bases from information models, etc.
- **Coding/Classification** - the ability to store and retrieve data, models, goals, requirements, specifications, designs or implementations based upon a "method defined" or "user defined" coding / classification scheme.
- **Automated Casebooks** - the ability to support the user with CAE instructions on how to use and apply all of the methodological tools provided by the organization.
- **Automated Environments** - the availability of an integration framework which allows and supports the use of multiple tools according to a development framework across the life of the application itself and over time in the integration program.
- **Tool Integration Support** - the ability to easily map data objects between the various models supported by the methodology. For example, be able to display all of the data objects that are associated with the data flow on a function model with the data objects on an information model.

- **System Architecture Design Analyzer** - the ability to model system architectures and identify the interfaces between the hardware and/or software components, and provide consistency checks and simulation capabilities to analyze performance characteristics of, for example, network response times.
- **Information Model Consistency Analyzer** - the ability to model complex constraints and assertions that can be analyzed through simulation. For example, assertions that affect data object state transitions, i.e., interdependency constraint rules.
- **Common User Interfaces** - availability of a consistent interface to tools for the system planner, analyst, designer, implementer, and maintainer. Much effort has gone into the design of consistent user interfaces for the end users. Little or no standards have been established for the tool vendors. Thus learning a new tool is difficult and time consuming. Simple presentation style interface standards as were developed by Apple Inc. for their MacIntosh software could prove very effective.

2.1.8 Technology Transfer Needs

As will be outlined in a subsequent section, training is only one phase of a major set of functions called technology transfer. Training fits into the educational process in that by definition training is intended to show people how to accomplish a task. During the past decade or more, training in the area of System Development Methodologies (SDM's) has been aimed at placing a set of system development methods, procedures and tools into the hands of the individuals who will be implementing them. In nearly every instance, training programs have been structured to fit into a lecture - workbook type of format. The typical student has been an individual from a company that has been assigned the task of learning the rules and syntax of the methodology. In most of the courses the student has been given a rule book or manual and has been provided with one or two days of lectures outlining how to apply these rules. For a certain class of students, and for a select group of SDM's, this format and approach to training has provided the students (users) with everything they needed to know. More recently, with new educational needs evolving, the old standard lecture - workbook format has not been effective for a broad enough group of students (users).

As the ICAM SDM's mature, so should the tools and materials used to train people in their usage. The SDM's are being refined and are no longer being used by just the engineers and system analysts. It is not only appropriate but required that individuals at many levels of the corporate organizational ladder receive the appropriate amounts of training in the application of the SDM's. Without a clear understanding of the tools, upper levels of management will be reluctant to accept proposals based upon the results of the application of SDM's. Within the engineering and system analysis ranks, appropriate training is needed to facilitate an effective application of the tools. In many companies, it is becoming common practice to provide appropriate levels of SDM training to persons who will be supplying data for the development of the tools and not necessarily performing the tool development itself.

The three areas of concern with regard to the existing training methods are:

- Training Materials
- Training Methods
- Training Personnel

As mentioned in the earlier discussion, current training materials for the SDM's consist of workbooks designed to be used in a lecture format. These materials were appropriate five to six years ago but may not be suitable in the future. Some attention should be given to the following concerns for the development of future training materials.

- An in-depth analysis of the materials to be used in training programs needs to be undertaken. The materials should not only teach the mechanics of applying an SDM, but should also provide some discussion of the foundation upon which the tool is based.
- A wide range of case studies illustrating the use of the SDM, and some discussion of the problems and benefits of such an application would be most appropriate. The current training material has limited case studies. A wide range of applications is needed.
- A lesson guide with case studies and working materials should be developed for a wide variety of students. For example, a class of managers does not need the same level of technical foundation of a tool as does the analyst. The lesson guide outline would provide an overview.
- A comprehensive workbook for each SDM, with the appropriate training tools for offering courses to managers, technicians and data providers needs to be developed.
- Videotaped presentations of successful applications of an SDM would be a significant addition to the training material.

Innovative training methods have historically not received a great deal of attention. One study performed four years ago under an ICAM project explored the possible use of programmed instruction and/or video tapes for use in SDM training. The results of this work indicated that a standard classroom lecture type format was preferred by students, and was the most cost effective means of presenting large volumes of information in a short time period. With the introduction of new systems development concepts and methods, some considerations need to be given to training procedures. For example:

- Possible uses of video tapes, video discs and some of the more contemporary training tools should be explored for integration with the more traditional lecture formats.

- Audience partitioning and audience identification procedures should be developed to assure a high level of technology transfer. A major problem in seminar presentation is the mix of the audience. Very often SDM seminars are attended by managers, technicians and non-technical people all at the same time. Procedures for handling this audience mix problem need to be addressed.
- A general set of rules for approaching an audience with SDM topics should be developed. Methods for introducing written materials to groups with varying interests needs to be explored further.
- Establishing an environment for SDM training is essential. Determination of the optimum medium, the size, type of room, and other environmental factors which influence the learning process should be explored.
- Use of "intelligent" tutor programs or expert modeling support environments which provide training as you use them should be investigated.

Many studies indicate that good training personnel are the key to effective training systems. People with an understanding of educational tools and no education in SDM would be weak teachers in an industry classroom environment. Technicians with strong SDM backgrounds and weak teaching skills would likewise not be successful in teaching SDM methods.

- Individuals with strong teaching backgrounds and the prerequisite tools for learning SDM should be provided with opportunities to learn and develop experience in using SDM tools.
- A team approach to teaching SDM should be explored. The team would be most successful if a group of industry educators could be brought together to form a presentation seminar team.

In many cases, training and education have been given only cursory attention on most methodology development projects. Training is the key to proper application of many of the tools. It is only through the development of appropriate training materials, procedures, and teaching personnel that the IISSE technology will be transferred from written manuals to the industry users. It is only through proper education of the management of these industry users that they will be afforded the opportunity to apply this technology.

2.1.9 Needs to Requirements to a Plan for Action

The results presented in this chapter represent the synthesis of reported industry symptoms and concerns as they have experienced shortfalls or failings in existing methods and tools within an information engineering initiative. The next section provides a summary of a

model of the "AS IS" information engineering process for system development. This model was used as a frame work for validation of the above described needs, and as a structure for assessing the state of the art in methods and tools. In Chapter 3 we report the requirements which evolved from these needs the results of the model analysis and the state of the art findings.

2.2 Models Of The System Development Process

As a part of this project, two draft models have been developed. They are identified as the Information System Evolution Process model (ISEP-0 — See Appendix B) and the Integrated Design System/System Development Process model (IDS/SDP-0 — See Appendix C). The ISEP-0 model is aimed at systematically describing how to build and use Information Systems Integration shells, such as an IDS shell. The ISEP-0 model appears in Appendix B of this report. The first section in Appendix B contains a node tree of the ISEP-0 model augmented with the mechanisms required to perform each function. The second section of Appendix B contains the actual IDEF0 model. The third section contains the glossary definition of each of the functions in the ISEP model. This ISEP model was developed as an extension of the original ICAM SDP0 model to serve as a generic model of the activities required to evolve an enterprise information system. There were two fundamental sources of experience for the concepts which are characterized in the ISEP-0 model. The first is based on experience gained from the IISS project. The IISS provided the basic architecture of the IDS and attempted to demonstrate how heterogeneous hardware and software systems could support the delivery of integrated data. The second source was provided by the IDS project itself. The IDS project is an application experiment which actually is building and implementing a prototype IDS using many of the concepts from the IISS testbed. The Rockwell IDS project involves both development and use of the IDS, and is expected to ultimately lead to a useful, generalized IDS shell. This model was then specialized (into the IDS/SDP-0) to highlight the characteristics of an IDS based information integration strategy.

The IDS/SDP-0, was developed to describe the process of assimilation, customization, and utilization of an IDS shell for the purpose of focusing the requirements assessment and state of the art analysis on the IDS Program and the use of its product. The IDS/SDP-0 model was developed down to the second/third level of decomposition with a detailed node index. Parts of the model are predicated on real experience, while other parts are an attempt to describe what is anticipated to occur where no real experience exists. The model is also an attempt to incorporate a multiplicity of views regarding how an IDS (or IDS-like) shell might be employed by an Enterprise to evolve, at its own pace, toward its own definition of a practical level of system integration. Viewed from this perspective, the IDS shell becomes simply an enabling technology to facilitate system integration.

The purpose of the ISEP-0 model is to provide an understanding of how to build and begin to use an information integration system, of which IDS is an instance. The IDS/SDP-0 model is aimed at understanding how to assimilate an IDS (or IDS-like) shell, customize it to fit an enterprise, and begin to use it.

The remainder of this section of the report presents the top level nodes of the IDS/SDP-0 model. In traditional IDEF-0 form, the A-2 through A-0 representations are used to establish a focus for the remainder of the model. It is readily apparent that an assimilation of this enabling technology is viewed as having strategic implications to the enterprise, and its implementation and evolving use is viewed as being managed/directed via a single, long-term

strategic program.

2.2.1 IDS/SDP-0 Model

This subsection contains the top level nodes of the IDS/SDP-0 function model which was used as a definition of the system development process within an IDS context. This definition was used to structure the state of the art assessment and the IISEE requirements definition. It was also used as a guide in the development of the IISEE plan. The complete IDS/SDP-0 model is included in Appendix C.

IDS/SDP-0 Model Purpose and Viewpoint

- The purpose is to represent the activity framework which can be used as a skeleton for planning how to evaluate, customize, and utilize (assimilate) the technology and capability comprising an IDS shell.
- The viewpoint is of the planner or manager responsible for introduction of IDS capability into the enterprise and the use of this capability in the best interests of the enterprise as a whole.

A-1 Manage Enterprise Strategic Development

The very nature of information integration vehicles such as the IDS implies strategic ramifications to an enterprise. While it is certainly true that IDS-like capabilities can be employed on a much more limited scale than as an enterprise-wide integration catalyst, experience with such devices appears to indicate that the greatest leverage might be gained from their employment throughout the enterprise as a whole as opposed to program or project confined utilization. This might, of course, require significant advances in technology over what is commonly (and affordably) available today. However, with their strategic potential having already been demonstrated, and with the requisite advances in technology apparently "in the wings," the decision to employ IDS-like vehicles is perceived as being justifiably strategic in nature, and the implementation effort is likely to be viewed as another of several on-going strategic projects in most enterprises in the future. The IDS/SDP-0 model attempts to address the implementation and use of an Information Integration shell, and an IDS shell in particular, from this strategic perspective.

A-0 Plan/Manage Implementation of Strategic Projects

Strategic projects are generally long-term efforts which contribute to strengthening the position of an enterprise with respect to its market and/or competition. They tend to be "sold" to executives based more on their potential benefits than on assurances of cost/benefit relationships since often neither the total cost of a strategic project nor its implementation

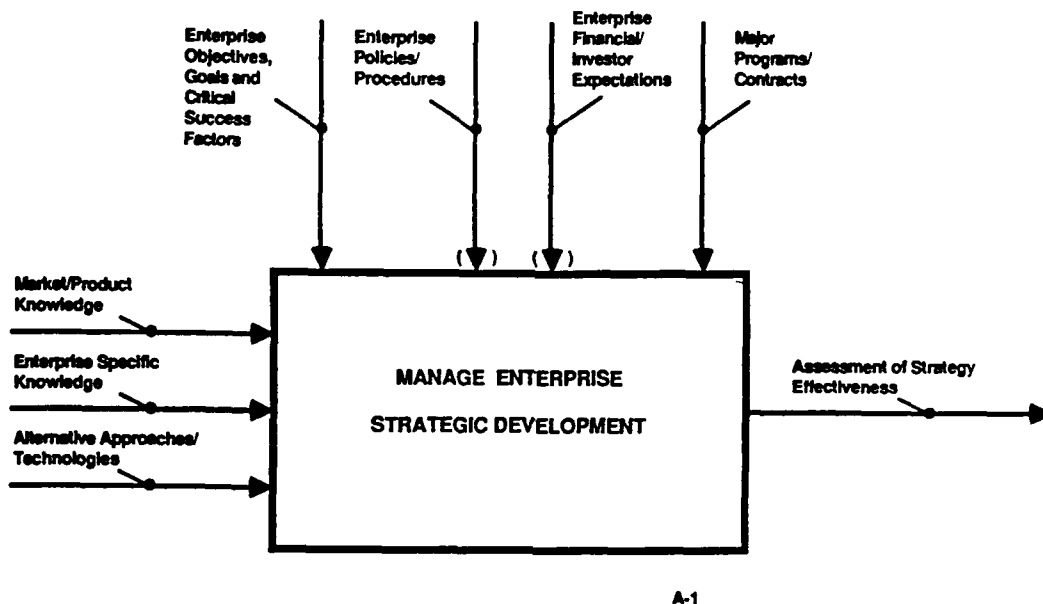


Figure 2.1: IDS/SDP-0 Context Overview (FEO)

schedule can be projected with any appreciable degree of certainty. Funding is generally provided on an incremental basis and predicated on a continuous stream of positive indications to enterprise executives that the effort warrants additional investment.

Implementation of an IDS-like capability is likely to be viewed by executives as a strategic effort; one of several that the executive body of the enterprise is pursuing. As such, it is likely to be directed and funded in a manner consistent with the handling of other strategic projects in the enterprise. The IDS/SDP-0 model reflects this assumption by treating an IDS implementation project as one "instance" of the strategic project portfolio of the enterprise.

Plan/Manage IDS Implementation Project

In the broadest terms, the implementation of the IDS-like capability in an enterprise is expected to track closely with the traditional life-cycle based administrative view of major projects, that is, to:

- Understand the problem
- Implement the solution
- Improve the solution

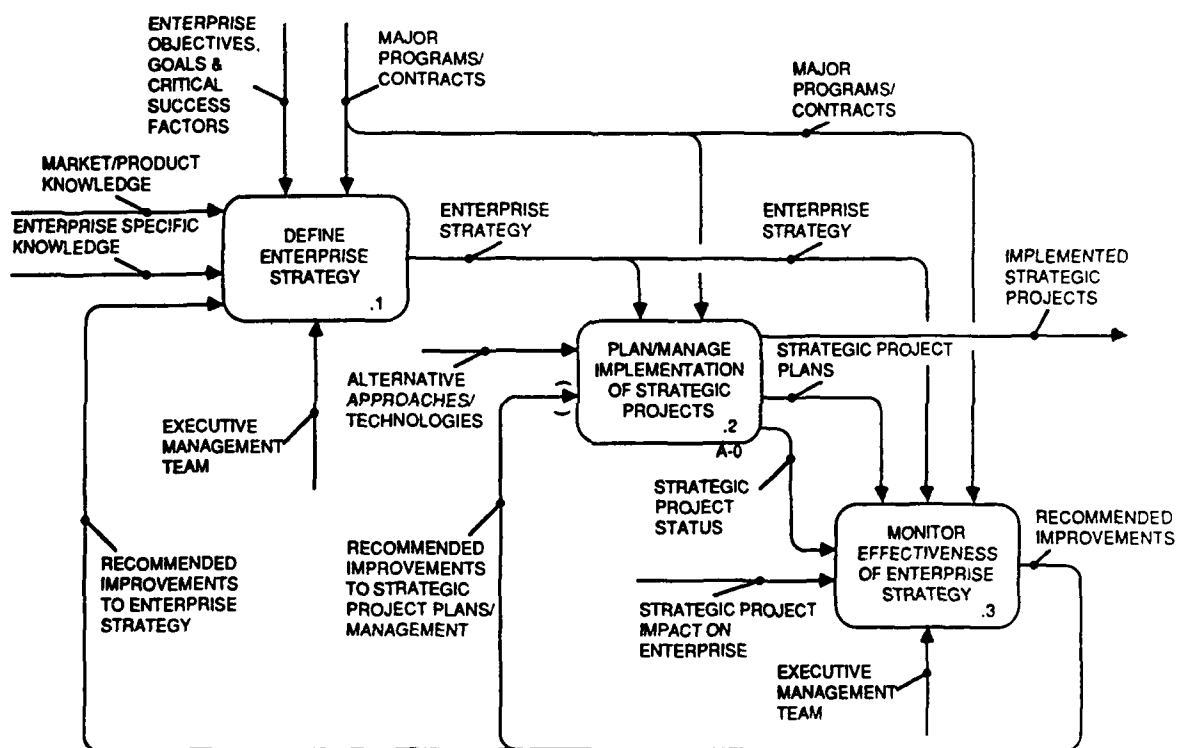


Figure 2.2: A-1 (Context) Manage Enterprise Strategic Development (FEO)

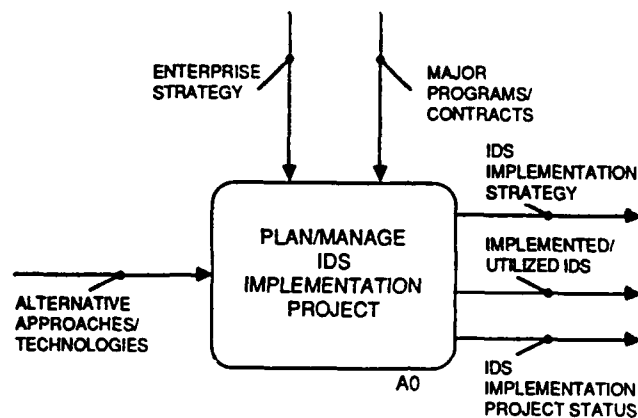


Figure 2.3: A-0 Plan/Manage Implementation of Strategic Projects (FEO)

However, several significant departures from tradition can be noted at the more detailed levels of the IDS/SDP-0 model, particularly in the technical portions (A22 and A23). This is primarily due to the perception of data as a shareable resource that is useful in many computer applications and by the use of recursive- and prototype-based approaches to the development and use of IDS-like capabilities.

Additionally, the IDS/SDP-0 model reflects the view that an IDS (or IDS-like) facility is really an enabling technology directed to achieve information integration in complex computerized environments, with the environment evolving slowly towards complete integration rather than quickly through one, large technology revolution. As a result, the IDS/SDP-0 model assumes continued refinement and tuning of the IDS technology itself during this evolutionary process.

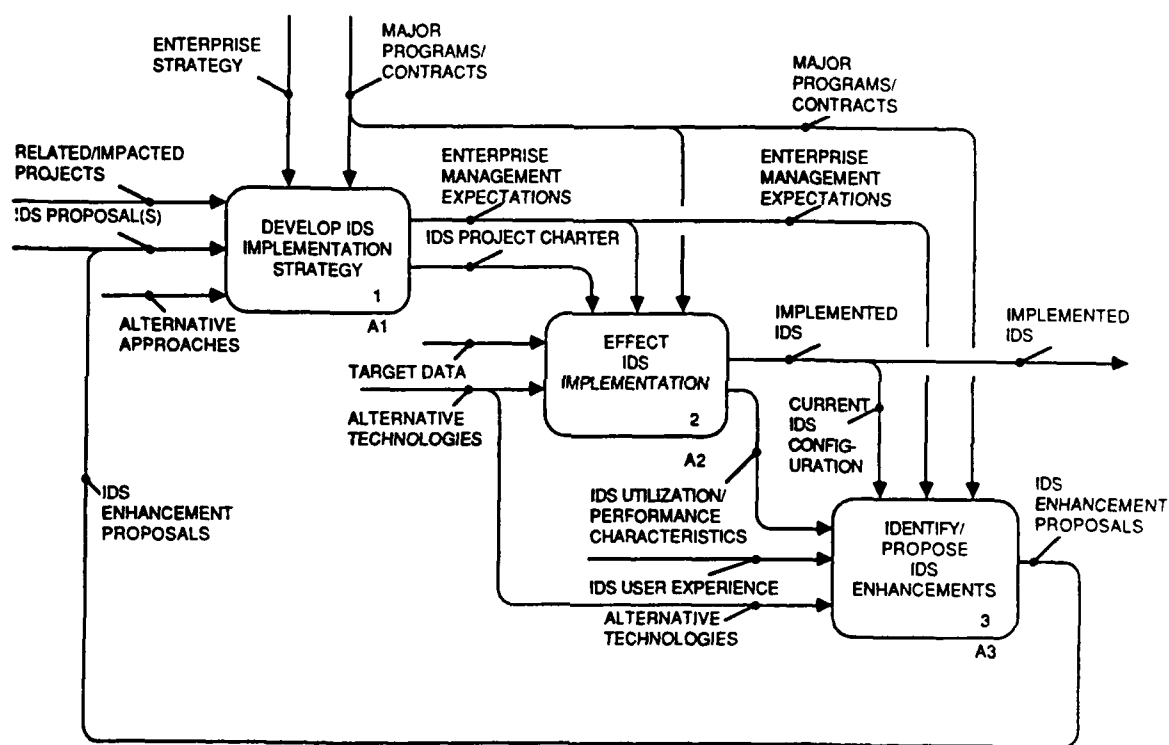


Figure 2.4: A-0 Plan/Manage IDS Implementation Project

2.3 State-of-the-Art Technology Assessment

The objective of this section of the report is to summarize the voids and deficiencies uncovered in the applied state-of-the-art (as opposed to emerging state-of-the-art) available technologies. In this case, technologies include both methods (practices) and tools supporting these methods. The context for the assessment was provided by three models of the Information System Development Process:

1. the SDM0 Model that emerged from ICAM Project Priority 1701, published in 1985.
2. the ISEP0 Model constructed in support of the ISEM Project (Reference Appendix B).
3. the IDS/SDP0 Model constructed during the current project for which this report has been prepared (See Appendix C). A description of the viewpoint and evolution of each of these models can be found in the introductory comments to Section 2.2 of this report.

2.3.1 IDEF0 Model Assessment

















Eighteen artifacts were identified as required for a complete technology assessment in support of an ISEE development the first nine of which were produced during this assessment project to prepare the ISEE plan:

1. A matrix displaying which mechanisms were identified for each of the leaf nodes of the SDM0 model (Appendix D).
2. A compilation of all the definitions and textual references in the SDM0 model for each of the mechanisms in the model (Appendix E).
3. A description of each kind of functionality envisioned as comprising, collectively, the mechanisms in the SDM0 Model (Appendix F).
4. A matrix displaying the composition of each mechanism in the SDM0 Model in terms of the functionality envisioned as comprising it (Appendix G).
5. A list of tools currently available and in active use which were envisioned as being representative of the state-of-the-art (Appendix H).
6. Tool Fact Sheet (1 for each selected tool) summarizing the characteristics of each tool as advertised by the tool manufacturer/provider (Appendix I).
7. A matrix displaying, for each selected tool, the functionality the tool is understood to provide and the typical usage of the tool in the System Life Cycle envisioned by the Manufacturer/Provider (Appendix J).

8. A matrix displaying the degree to which all the tools collectively provide each type of functionality overall is shown in Figure 2.5.
9. The IDS/SDP0 Model (Reference Section 2.2).
10. A matrix displaying which mechanisms were identified for each of the leaf nodes in the IDS/SDP0 model; similar to artifact #1, but based on the IDS/SDP0 Model instead of the SDM0 Model.
11. A compilation of all the definitions and textual references in the IDS/SDP0 Model for each of the mechanisms in that model; similar to artifact #2, but based on the IDS/SDP0 Model instead of the SDM0 model.
12. A description of each kind of functionality envisioned as comprising, collectively, the mechanisms of the IDS/SDP0 Model; similar to artifact #3, but based on the IDS/SDP0 Model instead of the SDM0 Model.
13. A matrix displaying the composition of each mechanism in the IDS/SDP0 Model in terms of the functionality envisioned as comprising it; similar to artifact #4, but based on the IDS/SDP0 Model instead of the SDM0 Model.
14. A list of tools currently available in active use which are envisioned as being representative of the state-of-the-art. These tools will form the basis of the final assessment conclusions. This artifact is an extension of artifact #5.
15. Tool fact sheets (1 for each selected tool) summarizing the characteristics of each tool as advertised by the tool manufacturer/provider. This artifact is an extension of artifact #6.
16. A matrix displaying, for each selected tool, the functionality of the tool is understood to provide and the typical usage of the tool in the System Development Life Cycle envisioned by the Manufacturer/Provider. This artifact is an extension of artifact #7.
17. A matrix displaying the degree to which all the tools collectively provide the functionality that appears to satisfy the envisioned needs of each IDS/SDP0 leaf node. This artifact is similar to artifact #8, but based on the IDS/SDP0 Model instead of the SDM0 Model.
18. A summary of observed strengths and weaknesses (voids/deficiencies) observed in the tools representing the state-of-the-art, insofar as they collectively provide the envisioned functionality required by the mechanism of the IDS/SDP0 Model.

The additional artifacts (items ten through eighteen above) are planned to be completed in the IISSE Requirements Project (see Section 3.5). The current assessment has yielded some indications of the following major deficiencies in the state-of-the-art technology base:

ARTIFACT 8 - COLLECTIVE RATING OF TOOLS BY FUNCTIONALITY CLASS

FUNCTIONALITY CLASSES	COLLECTIVE TOOL RATING
FUNCTION MODELING	
INFORMATION MODELING	
DYNAMICS MODELING	
STATE TRANSITION MODELING	
SPECIFICATION REPRESENTATION	
SIMULATION ANALYSIS	
CODING & CLASSIFICATION	
CONFIGURATION CONTROL	
PERFORM. ANALYSIS	
LC ARTIFACT MANAGEMENT	
DATABASE GENERATION	
SOFTWARE GENERATION	
ECONOMETRIC ANALYSIS	
CLUSTER ANALYSIS	
SURVEY SUPPORT	
TEST CASE GENERATION	

FACE LEGEND





-  - VERY WELL COVERED; NO MAJOR VOIDS/DEFICIENCIES PERCEIVED
-  - GENERALLY SATISFACTORY, WITH RESERVATIONS; NO MAJOR VOIDS, BUT SOME DEFICIENCIES PERCEIVED
-  - GENERALLY ADEQUATE, BUT WITH STRONG RESERVATIONS. SOME SIGNIFICANT VOIDS/DEFICIENCIES PERCEIVED
-  - GENERALLY POOR; MAJOR VOIDS/DEFICIENCIES PERCEIVED

Figure 2.5: Tool Functionality Matrix

1. Methods and tools for determining information strategies (as opposed to application development strategies) do not appear to be available from a broad selection of suppliers. An information strategy is the requisite foundation upon which successful assimilation of IDS and IDS-like facilities is based. Those methodologies and tools purporting to provide an information strategy appear to have their roots in application development planning, and generally subordinate their address of information to its use within a given application area. Even though information requirements are generally identified from an enterprise-wide perspective, and it is readily evident that the same kind of information must be shared by multiple processes/applications in the enterprise, the information model construction appears to generally be planned on an application by application basis.
2. Five notable deficiencies appear to exist:
 - (a) It is generally unclear what components of the enterprise strategy influence the composition of the information strategy, and in what manner that influence is reflected.
 - (b) No clear mechanism exists for determining the relative importance of one kind of information versus another, from the enterprise viewpoint.
 - (c) No clear mechanism exists for determining the order in which information models should be developed as a foundation for the conceptual schema of an IDS-like facility.
 - (d) No clear mechanism exists for determining the best order of application development, based on the relative importance of information from the enterprise perspective.
 - (e) No clear mechanism exists for extracting an evolutionary (as opposed to revolutionary) migration plan from the information strategy that optimizes the use of existing information assets via an IDS-like facility.
3. Commercially available System Development Methodologies (SDM's) represent frameworks for building and maintaining stand alone information systems. They are overwhelmingly application project oriented. Although not identified specifically in the list of tools (since tools are viewed in this context as software), none of the most commercially popular SDM's that were reviewed appeared to require the use of anything resembling an IDS-like facility, nor did they require/facilitate the development/use of common data standards (conceptual schema) for shared data from the Enterprise perspective. Some, however, did allow for the development of information models targeted for specific applications.
4. Another major deficiency of SDM's in general appears to be the absence of any effective means for assuring correct and non-conflicting requirements, leading to significant

maintenance costs built into the resultant applications. In general, methodology frameworks which address Information Resource Management issues, philosophies, styles, etc. from an enterprise perspective appear to be almost non-existent.

5. In the aggregate, the claims made by tool manufacturer/suppliers might lead one to assume that all aspects of systems development are adequately addressed. There are, in fact, many tools available, and it does appear that every major aspect of the System Development Life Cycle is addressed by one or more tools, from strategic information planning to maintenance. However, an assessment of what the tools actually do and what the consumer-base (tool-users) is actually experiencing, leads one to a somewhat different conclusion. In addition to the issues already discussed under items 1 & 2, for example, there appear to be serious deficiencies in supporting the System Design Process. Most tools claiming to provide design support appear to do a much better job of supporting analysis. While these tools aid in the capture of facts upon which a design can be predicated, and further aid in capturing the design itself, few (if any) provide much support in the actual derivation of a design. The process of design appears to remain an overwhelmingly human function.
6. In terms of functionality provided (as defined in Appendix F) some notable voids and deficiencies appear to exist, as follows:
 - (a) The apparent absence of a common, unified, theoretic foundation for modeling tools as a class of objects.
 - (b) The predominance (in terms of availability) of modeling tools that capture pictures (bit map storage/retrieval) as opposed to those that capture facts and generate pictures (models) from those facts.
 - (c) The rarity of tools that address:
 - i. Entity State Transformation
 - ii. Information System Simulation
 - iii. Coding and Classification of Information system components; e.g., data, software, etc.
 - iv. Configuration management of system components; e.g., schemas, schema views/projections, software, etc.
 - v. Life cycle artifact management; e.g., function models, information models, requirements, specifications, etc.
 - vi. Database design (as opposed to simple information model transliteration)
 - vii. System/information econometric analysis
 - viii. Cluster analysis (of system/information components)
 - ix. Survey support

- x. Test case generation (as opposed to test data generation)
 - xi. Specification generation (from requirements models)
7. Some of the currently perceived voids/deficiencies may be due in part to the limited sampling of tools. It is entirely possible that a broader search would enable the discovery of tool that adequately address many of these issues.

2.3.2 On to Concept

The needs, state of the art voids, and the definition of the IDS system development process were used as a basis to formulate a preview of the architecture of an IISSE. This architecture is presented in the next section. It is an initial attempt by the coalition to identify the form and structure of a solution to the information engineering framework, method and tool needs. Most importantly it was developed to serve as a structure for development of the tactical plan for filling the identified needs and voids.

2.4 Conceptual Design of Structured Methodology

The purpose of this section is to provide a description of the requirements for an IISEE based on company needs, the development process definition, and the state-of-the-art review results. This section also contains a high level architectural view of an IISEE and an overview of its intended philosophy of use.

2.4.1 Characteristics and Requirements

The requirements for an IISEE as identified by the coalition team associated with this effort can be summarized as follows:

1. Complete Life Cycle Coverage (from strategic planning through maintenance).
2. Focus on three schema architecture information integration strategy as defined in the IDS concept.
3. Support the construction of the three schema implementation mechanisms (for those who want to build their own) and the installation and "feeding" of an existing mechanism (for those who want to implement the IDS product directly).
4. Adapt to an organizational "system" context (i.e. accommodate existing organizational methods and system environments).
5. Adapt to a dynamic company environment.
6. Provide methodology and tool integration mechanisms.
7. Provide methodology, tool, and implementation integration standards.
8. Provide methods for capture and management of evolving target implementation environment requirements (known as "design-to" requirements).
9. Provide automated support for maintenance data collection.
10. Demonstrate faster development time and overall lower life cycle costs through life cycle artifact control.
11. Provide continuous migration paths for the information system applications.
12. Provide support for the elicitation of knowledge / system description information / needs / requirements from the user community.
13. Allow for extensions to the framework, methods, tools, and environments.

14. Provide the capability to educate users in the limitation of an existing system and the implications of stressing the system.
15. Allow modularity in the development approach.
16. Provide tools to support the organization of needs, plans, requirements, and designs.
17. Provide expert system capabilities to guide technical and managerial users in the execution of all steps within a framework.
18. Support the isolation of business / engineering / manufacturing / logistics logic and data from implementation technology.
19. Provide sufficient capture of scoping, requirements, design, and implementation decision making rationale to minimize impact of personnel changes in the project team.

Taken together these requirements for an IISEE imply that what is needed includes not only a complete set of frameworks, methods, procedures, environments, and tools; but also an *IDS*-like environment to allow system developers to share and control the evolving system development data.

2.4.2 Description of the IISEE Concept

Much of the IISEE concept can be understood by consideration of the name itself. The first word *Integrated* is meant to imply that the set of components (frameworks, methods, development procedures, computerized environments, and tools) are design to fit together in such a way as to reduce the effort required to produce integrated information systems. The second two words *Information System* specify the kinds of systems the IISEE is intended to address. Thus, the product might require excessive work for a simple, stand alone application. It would also be inappropriate for a specific material handling system development, or a specialized weapons system itself. The fourth word *Evolution* is meant to convey the fact that the IISEE will be set up to be:

1. Used over the life of the organization. The IISEE must be expandable to support the addition of new methods and tools as they are developed or as new technology dictates.
2. Inserted into an existing company setting. It will take into account that in the typical application the organization will be trying to evolve *existing* systems into an integrated system, not just replace existing systems with new systems.

The last word *Environment* implies that the frameworks, methods, and development procedures must be complimented by a set of automated support tools integrated into an automated environment which maintains the "critical" life cycle artifact ¹

¹The term "artifact" used in this context refers to information or data produced as a part of the system development process. It is analogous to the term "product data" in the *IDS* world.

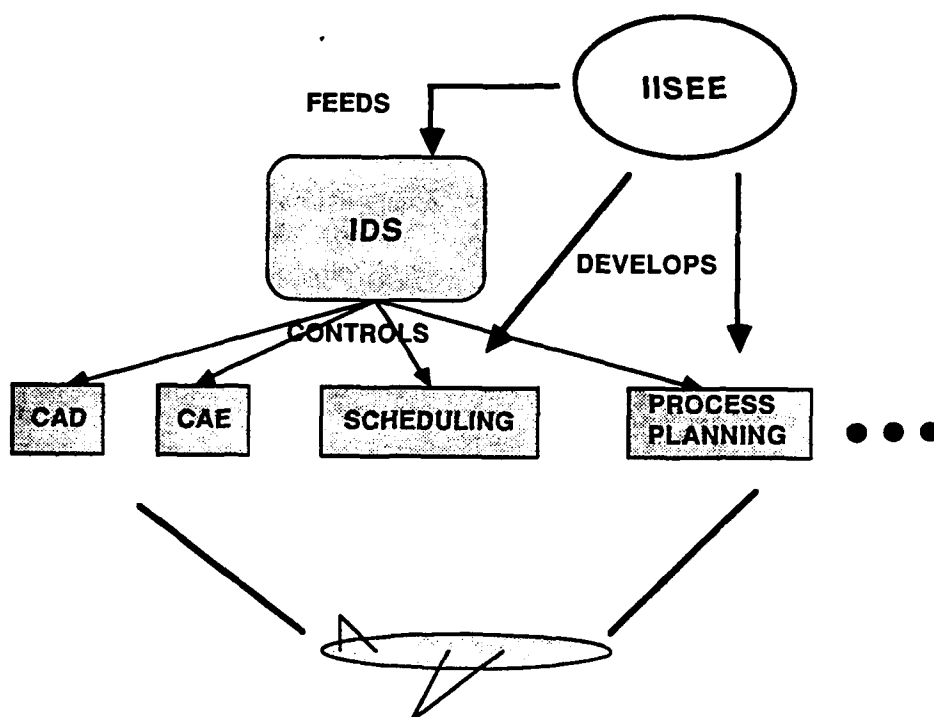


Figure 2.6: Relationship of IISEE to the Information and Product Systems

The IISEE is itself a system. It is a system which is used to evolve an integrated information system for an organization. The information system which is evolved through the use of the IISEE is what is used to design, build, and maintain the weapons system product. This complex set of relationships is illustrated in Figure 2.6. It is important to note that the IISEE is intended to be designed to be insensitive to the "changing out" of the shaded systems in that figure.

2.4.3 IISEE Architecture

The IISEE structure as currently envisioned is illustrated in Figure 2.7. One of the most important components in the IISEE is the "Guidebook." The Guidebook contains a description of both the management and technical development philosophies which are "assumptions" in the Framework components. In other words, if one wishes to know "why" a particular step is required, or why the sequence of steps in a Framework is the way it is, then the answer should be found in the "Guidebook." Thus, the IISEE is designed to carry with it its own design rationale! The coalition associated with this project feels that without such rationale there is little chance of such a product being accepted into widespread use.

The Guidebook would also contain information on how to "size up" a particular de-

velopment situation (i.e. what landmarks to look for), and how to choose an appropriate Framework for the situation. The complexity of making such a decision is the reason for the need (identified by the coalition) to accompany the Guidebook with an expert system which would assist in such decision making. There is another intended function for this knowledge based system: to help the user when he finds that his initial selection of a framework was incorrect and he must switch between frameworks. This could happen in the middle of a project or could be a natural result of the evolution of the organization over time.

The next major component of the IISEE is the collection of "frameworks". A framework can be thought of as being analogous to a "script" or even a recipe. The framework contains a "Development Procedure," a definition of the organizational skills that will be required, and the particular "Methods" chosen from a set of methods which will be utilized in the various steps of the Development Procedure. It is anticipated that there will probably be specialized "Method Use" procedures associated with a Framework which specialize the use of the results of a particular method in the context of the framework. The Development Procedure will be broken into phases, steps, and tasks. The tasks describe specific activities which must be undertaken by the participants, the steps represent significant technical decision points, and the phases indicate key management decision points and milestones.

The next major component of the IISEE is the "System / Software Factory" (SSF). The purpose for prepending the term *System* is to emphasize the point that the computerized environment and tools contained therein will eventually cover all of the activities from strategic planning through maintenance of the system. The utilities contained in the "Computerized System Development Environment" (SDE) component of the IISEE are described in Section 3.3 of this report. Ultimately, the "tool crib" of this factory will contain a complete set of power tools for the system developers.

The last major component of the IISEE is the "Technology Transfer" mechanisms which must be put in place. These components are important not only to get the IISEE into the organization in the first place, but also to propagate its use throughout the organization. A description of the needed elements of this component is contained in Section 3.4 of this report.

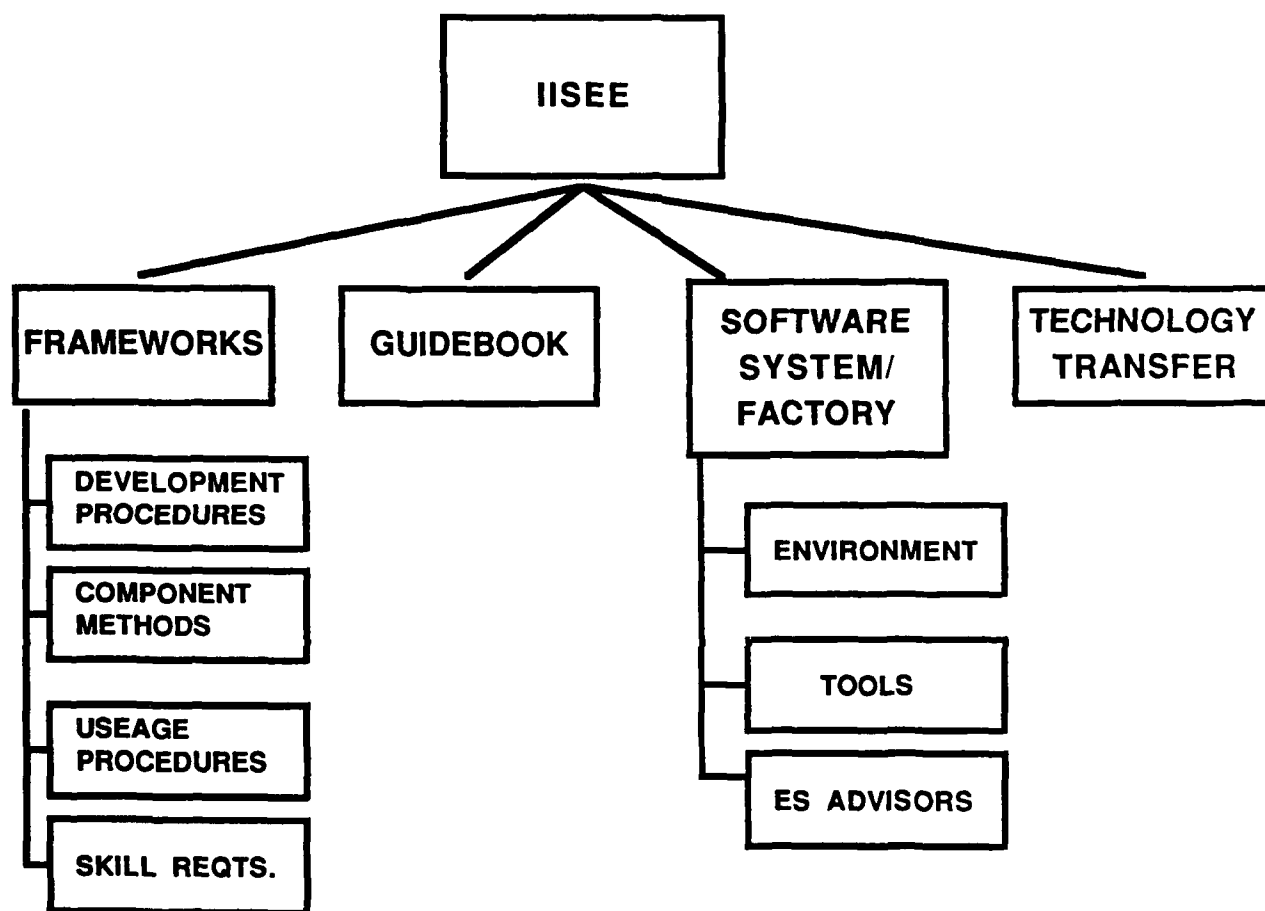


Figure 2.7: IISEE Product Structure

Chapter 3

Plan for Development of Structured Frameworks

The purpose of this chapter is twofold. The first objective is to summarize the requirements for an IISEE. The second is to provide a plan for IISEE development. This plan (if carried out) will provide the evolution of an enabling technology which will (with existing available methodologies and automated tools) provide an IISEE for organizations to use to implement and maintain information integrated engineering and manufacturing systems. Section 3.1 presents a summary justification for the IISEE component and framework development within the context of IDS. Section 3.2 identifies the requirements for framework and component method developments for an IISEE. Section 3.3 identifies requirements for automated tools and environments to support the application of the framework and component methods. Section 3.4 identifies the technology transfer efforts required to support the dissemination of the results of this effort and facilitate industry acceptance and usage. Section 3.5 provides a series of project descriptions, with time and resource estimates, for the development of the enabling technologies needed to fulfill the requirements identified in the sections 3.1 through 3.4.

3.1 IISSE Program Plan Rationale

The purpose of this section is to provide a summary rationale for proposed IISSE developmental projects. The projects in the actual plan itself are organized around technical thrust areas (see section 3.5). This organization is preferable for program management purposes, but it tends to obscure the logic and direction behind the IISSE. This section is meant to clarify those issues.

One of the first activities of this project was to acquire as much information as was possible on a previous project, "Systems Engineering Methodology" (SEM), which was a part of the Air Force Integrated Computer Aided Manufacturing (ICAM) Program. It was hoped that the architectures from this project and much of the thinking which went into the component methods tools and environments from this program could be directly built on the IISSE. As is evident from the method and tool state of the art study (see chapter 2 and most of the appendices), we also desired the IISSE to build on the current state of the art in commercially available methods and tools.

While to a large extent our intuitions were correct, there appeared to be an obvious shift in logic behind the systems which those earlier programs and commercial products were meant to service and the systems that the IDS program has envisioned. The emphasis of most of the existing tools and the SDP models of the SEM project was on **application engineering**. But the IDS, and the industry in general, is moving away from the "one at a time" applications engineering toward a new approach where applications are viewed as a part of an environment. The overriding theme of this new concept is "common data standards" and the sharing of common data. This is not a revolutionary concept as information integration thrusts emerged over ten years ago. The recognition by industry of the need for a shift in logic relative to the methodology by which information systems are evolved is emerging.

Just as John Zachman applied the similarity between architectures for buildings and information systems architectures, a useful parallelism can be drawn between this new shift in logic and community planning. The concept of community planning revolves around the notion that there are certain elements of our environment which affect the entire community. These elements are the responsibility of the entire community. The effect of this recognition was the organization of community standards. These standards changed the design and development approaches which can be used in the community (even to the point of constraining the architectural alternatives of the designer of buildings in that area). A similar impact exists in the information engineering approach to applications development within an IDS. No longer can all of the decisions regarding an applications structure and use of data be made in isolation. The new paradigm requires that the application developer have access to a framework for structuring his activities in the context of the overall information integration program.

With these concepts in mind the IISSE development plan (see Section 3.5 of this report) evolved to address the following critical questions:

1. What are the critical methods for information engineering?
2. What is an IDS architecture?
 - (a) What is the critical functionality?
 - (b) What are the information and technology concepts and implications?
3. What is the integration framework methodology for:
 - (a) Implementing an IDS architecture in my environment?
 - (b) Customizing the Rockwell architecture?
 - (c) Evolving applications in an IDS architecture?
4. What is an IISSE and how does it relate to the IDS architecture?
5. What technology will the IISSE provide for planning and management of an IDS implementation program?
6. What technology will the IISSE provide for project management in the IDS environment.

Sections 3.2 through 3.4 provide a summarization of the requirements implied by these questions, the needs analysis activities, and the TAP inputs. Section 3.5 contains the initial IISSE program plan which addresses these needs and requirements. It is expected that this plan will evolve considerably between now and its eventual funding, but every journey begins with a single step.

3.2 Component Methods and Framework Recommendations

The following subsections summarize the framework and component methods requirements which are addressed in the IISEE Program plan presented in Section 3.5 of this report.

3.2.1 Framework Requirements

There are several ways to categorize the needed integration framework developments. Our justification and rationale is based upon analysis of the needs and voids described in Chapter 2 of this report. The following are the target system application types which must be supported by the integration framework developments followed by the required organization profiles which must be supported.

1. Target system application types:

- (a) CAD / CAE system development
- (b) Engineering Data Management and Control system development
- (c) Three schema architecture based information integration support system development.
- (d) etc.

2. Target user organization size and profiles:

- (a) Large DoD prime contractors
- (b) First tier DoD subcontractors
- (c) Second tier DoD subcontractors
- (d) Large commercial corporations
- (e) Small businesses
- (f) Systems consulting organizations

3.2.2 Extensions to Existing Component Methods

Most of the recommendations associated with the existing methods center around the following three areas:

- 1. Formalization of definition and syntax.
- 2. Provision for improved training materials and mechanisms.
- 3. Development of the "Use" procedures (i.e. how to use the results of a method application once you have invested in those results).

3.2.3 New Component Methods

Even with the large number of available methods in the market place, the experience of the coalition team associated with this project and the ongoing IDS project identified a number of voids which need to be filled. The following list identifies a number of methods which must be developed to fill technological and operational voids:

1. Activity state models
2. Data state models
3. Data object flow timing descriptions
4. Process flow timing descriptions
5. Organization structure and flow of control models
6. Constraint specification rule languages
7. Domain specification languages
8. System architecture models
9. Implementation environment models
10. Organization objectives and goals alignment methods
11. Information and business strategy alignment methods
12. Data costing methods
13. Project prioritization methods
14. Project costing methods
15. Method for information integration of methods
16. Mechanism (function, structure, and performance) models
17. User interface models
18. System design logic models
19. Schedule modeling relationships between data and activity objects methods
20. Computer system architecture describing and modeling methods
21. Conceptual to internal schema mapping methods

22. Design of external schemas and external to conceptual schema mapping methods

A general requirement on any new component method is that where possible it should be designed as an extension to an existing method. Another general requirement on any method development is that the formalism for that method be developed in conjunction with the discipline and use components. These requirements have been mapped into a series of development projects under the "Component Methods" thrust in Section 3.5 of this report.

3.3 Computerized Environment / Tool Requirements

Existing commercial tools address, to varying extents, many of the functionality classes needed to support an IDS-like system development process (see Section 2.4). Specific voids within these classes are relatively straight forward to identify and advanced plans for methodological completion can be constructed. However, the overriding flaw that permeates available support tools is the complete lack of an integrating environment, and a unifying prescriptive methodology for integrating environments and tools. The situation is akin to carefully putting together a design team for a project, and then locking each member in a room with no means of communication. Because of the importance of the environmental question, this section is organized along the lines of environmental support versus specific component tool support.

3.3.1 Integrated Computer Support Environment Requirements

An overall model of an adequate environment includes components that are targeted for development and components targeted for operation. There must be communication between the two, and there will be some overlap. However, an idealized model can be visualized in Figure 3.1. Such a System Development Environment (SDE) integrates tools and their work products across the entire development life cycle. A meaningful integrated environment must provide a uniform user interface. Ultimately it provides a conceptual framework for the user. A methodology must be provided for controlled evolution of the environment as hardware, software, and methodologies change. The environment should support team interaction among developers with electronic means of visual communication and electronic dialog. The embodying methodology of the environment should support both top-down and bottom-up development, constraints, software development organization, and system management. For productivity issues, the environment should support reusability. A cataloging and retrieval mechanism is necessary to accomplish this goal.

Hardware / Operating System

The most visible characteristic of the SDE has to do with hardware support, which is no more homogeneous than the tasks to be performed in such a comprehensive and ongoing effort. The following items are an indication of the range of hardware support that is needed.

Networking

Networking capabilities that will allow communication among a multitude of dissimilar machines is a necessity. Further, such communication must not be restricted to file transfers, but must also include communication among databases and processes.

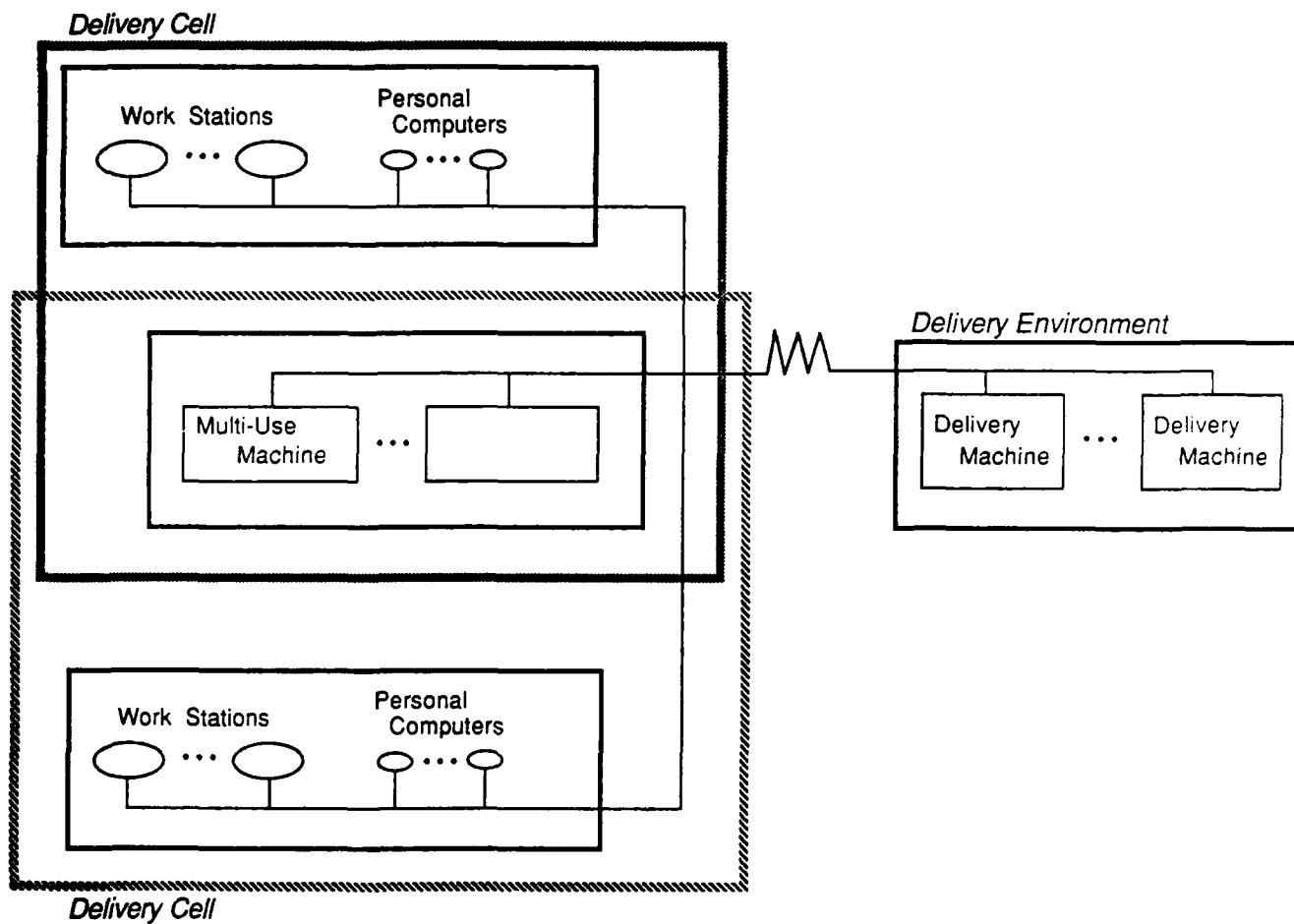


Figure 3.1: Ideal Environment Model

Window Support

The much touted capability for windowing is not overly critical in this context, particularly in the development phases. It is a gross waste of human effort to expect a developer to work with a complex, multifaceted system, and to only be able to see one aspect at a time. A simple example is the ability to look at a section of source code in one window while viewing the execution of the code in a second window. Adherence to the X-window standard is recommended.

Shared Development Hardware (specialized e.g. Lisp machines, workstations)

The Lisp machines have come to be popularly identified as Artificial Intelligence Workstations. However, the history of work in AI has somewhat fortuitously yielded a by-product of tremendous potential for IDS-like projects - the Lisp machine environment. This environment has something to do with the Lisp language, but a great deal more to do with a well planned work environment. The first underlying theme is that the environment tracks and "understands" activity in a number of cooperating processes which can be operating conceptually in parallel. Thus, for example, compilation can take place directly from one or more editor buffers, mail can be sent, documentation can be inspected, and many other activities can take place as the user sees fit to move among them. The second underlying theme is that the coordination scheme is for the benefit of one user who must make progress on many related fronts at once. Thus, the machine type is normally a one user at a time arrangement, although a small group of people may actually utilize the machine in sequence. The ability to tailor individualized "worlds" facilitates the sequential use of the machine by multiple users.

The increase in developer productivity, particularly during the design and engineering phases when prototyping cycles are the rule, is so substantial that efforts to better integrate the hardware type with more traditional types should be required.

Personal Computer Hardware

Personal computer hardware definitely has a place in the process, both as a workspace for an individual to try out ideas and as a communication device within the overall system. Mobility and immediate availability are primary assets of the personal computer.

Multi-user Hardware

Multi-user hardware in general refers to the class of mini and mainframes that are industry workhorses. For tasks (such as coding of large systems) that must proceed in parallel and are well defined, they are the machine of choice. The multi-user characteristic is critical in providing for configuration control.

Delivery Hardware

Delivery hardware must normally have the multi-user characteristic. Of course, physical characteristics that affect execution speed and storage access become critical in this environment. The major point to be noted is that the characteristics that make hardware fit for a delivery vehicle are quite different than those needed during development, although the same machines have traditionally been used for both development and delivery.

User Support

One goal of an integrated environment is to support the user with as much automation as possible. Clerical and mundane tasks should be machine mediated. The machine should be able to supply the user with successive steps from the applicable integrating framework and assistance both in performing these steps in their work activity on demand, and in maintaining the status of his progress against the task set.

User support issues run throughout the development and delivery environment, and are emphasized in the hardware section. Nevertheless, several user support characteristics that are specifically in the software domain can be identified.

Common User Interfaces

User interfaces should have a common style, regardless of life cycle phase, in terms of such aspects as keystroke sequences, use of menus, command names, method of accessing background information, and method of moving around in the environment. The benefit of a common style is evidenced by the success of the Apple MacIntosh interfaces. The commonality of program interaction renders paper copy of system documentation virtually superfluous.

User Profiling Capability

The user profiling capability is important in the development stage to enhance the efficiency of developers by allowing for individually tailored environments and to aid in project management by monitoring and controlling access to project components. It is important both to control access to information and to allow for the formulation of individualized query responses.

Online Documentation

Documentation of system functions and system software should be readily available online. Additionally, at least the syntax and procedures associated with specific methods that are being used should be well documented online. Access should be provided by choice of topical index and information at progressively greater levels of detail should be provided.

Life Cycle Artifact Management

One of the most often discussed issues of system development is the failure of a project to capture the corporate history of system development. This shortcoming is particularly obvious during the maintenance phase. All artifacts, including requirements, specifications, designs, associated documentation and interaction over such artifacts needs to be captured. Requirements traceability would be enhanced if the LCA management system properly related system artifacts.

Configuration Management Utilities

Configuration management is that part of the life cycle artifact management system that handles versions and change control of emerging and evolving system artifacts. The underlying notion is that artifacts must be identified, collected, and controlled in order to assure proper distribution of the finished system and its supporting work products, as well as to assist in evolution of the product.

Integration Utilities (inter-methodology, intra-methodology)

Integration utilities assure smooth transitions across the life cycle phases for both the developer and artifact being developed. For instance, the specification methodology should interface cleanly with the design methodology. That is, the output of the specification methodology should be the input driver for the design methodology.

3.3.2 Component Tool Requirements

Component tools are those individual tools that support specific functions within the systems development process. They should be used within an environment that is cognizant of their functional role and supportive of tool integration. For each method in the IISEE framework a component tool is needed for:

1. Support of data collection
2. Support of model representation generation and display
3. Storage/retrieval/printing of models
4. Integration of method results into the Life Cycle Artifact database

Based upon the results of the state-of-the-art review presented in 2.3 there are many component tools which are already commercially available. To use these tools in an IISEE automated environment would at a minimum require:

1. The definition of interface standards

2. The integration of the tool concepts into the IISSE Conceptual Schema

As can be seen from the survey results, many existing tools do not enforce rule methodology. Rather, they provide drafting and text management support. The strategy for component tools that we are recommending is as follows:

1. Develop a set of modeling tool generation utilities which can be used to build new tools.
2. Define a set of data exchange standards with existing tool vendors so that existing tools can be integrated into an IISSE structure.
3. Using the modeling tool generation utilities, develop prototype tools for the following high priority application areas where no existing tools are available.
 - (a) Data state modeling
 - (b) Process modeling
 - (c) System architecture capture and simulation
 - (d) Coding, classification, and cluster analysis for:
 - i. needs
 - ii. requirements
 - iii. system components
 - (e) Conceptual Schema design generation from information models
 - (f) Rapid prototyping of external schemas
 - (g) Workstation based data collection and summarization tool
 - (h) Rapid prototyping of IDS applications

In addition to the above tool recommendations there are a number of additional tools which require the application of expert system technology. These will be presented in the following section.

3.3.3 Knowledge Based Tool Requirements

Applications for Knowledge Based Systems (KBS), which were identified by the project, may be generally classified as those applications which support an IDS development cycle, and those which augment the IDS capabilities. That is, those which aid in building / maintaining an IDS based environment and those which would be part of the IDS itself. The emphasis in this section is on knowledge based development support tools since the latter tend to be domain specific.

1. Modeling support is a major area that could benefit from KBS support. The model building tools on the market (primarily PC based) have mainly addressed the syntax, and to some extent the procedural components of a method. Generally, a tool has no notion of the underlying concepts and theory of the method or of the reasonable usage of the method. Thus, it neither enforces the theory nor anticipates appropriate model use. This is a fertile area for knowledge based augmentations to the model building process. Samples of rule based buttressing that can be envisioned are:

- (a) The method semantics that govern the modeling process should be enforced.¹ Otherwise, there is no way of assuring that the method that is administratively espoused is the one actually used. An information model may, for instance, easily have the visual appearance of an IDEF1X model without being an IDEF1X model. Actually, some tools on the market do offer semantic support for some types of models (the AutoIDEF0 tool, for example). However, the more common case is that drafting tools with little or no consistency checking are all that is available.
- (b) The usage of a viable model will continue long after its completion. Certainly query facilities for the IISSE in a natural language (see Chapter 5) format, or perhaps in a graphical pattern matching format, should be provided.
- (c) Construction of the necessary models required to support planning, analysis, design and construction and evolution of an IDS implementation will require a level of organizational participation which is far in excess of that supportable by trained modelers. A model data acquisition and model prototyping support utility base on use of natural language discourse processing is required to allow the domain experts to more productively contribute to the definition effort.
- (d) Much of the information contained in a model will be useful in other contexts - models, programs, and databases. Systems able to intelligently extract information from a model that is relevant in another context would be extremely valuable. Today, a large systems development project will likely include at least the construction of function models, information models, process models, and data structure models; but the relationships among the models are manually derived

¹The question of enforcement of model semantics is complicated by the consideration of the various usage modes in which a person may do modeling. One major usage differentiation has to do with the scope of the modeling effort; that is, a person will typically approach the problem differently depending on whether one is building a small, "personal" model or is contributing to a large scale company model. An effective tool must be able to differentiate between these modes. A second major differentiation has to do with the phase of the modeling effort underway. During the early phases, it is unreasonable to insist on completeness and absolute consistency. Rather, inconsistencies should be flagged at appropriate points. Also, in some methods (IDEF1 for instance), the syntax and semantics that is allowable in early phases may differ from the final correct syntax and semantics. The life of a model does not end with the completion of its construction. Actually, a model, like software, may continue to slowly evolve long after its initial "completion." Thus, a maintenance mode of operation should be provided.

by modeling experts - if such are available. Not only are such people scarce, but also their knowledge goes with them when they leave the project.

- (e) It was pointed out in Section 1.1 that a method may have multiple presentations, each of which may be useful depending on the orientation of the viewer. A natural use for a knowledge based system would be the formulation of a model in its various guises.
 - (f) It was also pointed out in Section 1.1 that a methodology usually encompasses a family of related methods, e.g. both ENALIM and IDEF1 are methods within the information modeling methodology. The "translation" of a model associated with one method to a model in the style of another method is far more difficult than generation of multiple presentations, since the information content of the two will not be exactly the same. A heuristic solution to the problem seems to be the most promising approach, since it is possible to identify rules of thumb for converting some constructs within one model to corresponding constructs in another. Part of such a system would be the selective prompting of a user to direct attention to problem areas. That is, it may be easier to identify a problem than to correct it. In fact, it may not be possible to correct a problem automatically because of an information or relational gap.
 - (g) Validation of a model could be supported by a system to either generate an English statement of the business rules that are implied by a model, or to incorporate statements of business rules into a model.
2. The long range view of the IISEE project includes an expectation that a classification structure of environment types will be identified. For example, the implementation of an IDS-like system in the context of a flexible manufacturing cell would imply a certain type of development procedure. Given the type identification, a development framework must then be configured for an individual customer. An expert system that utilized a question and answer format to customize a framework for a company is required. Of course, this suggestion anticipates the development of human expertise in this area. The question and answer format would appear to be necessary in order to gather specific company or development personnel information. The question of acceptable risk will be highly company and project- and time-specific.
 3. The implementation of an IDS-like system implies a substantial learning process on the part of the team members. In particular, selected methods and method presentations must be learned. Applicable standards must also be learned by project personnel. Intelligent tutorials that accommodate the learning style of the user could greatly speed this process. The observation that most modeling efforts involve a classification activity leads to one major contribution of such tutorials - the provision of practice / critique cycles, which are needed because it simply takes practice to teach someone to

be good at classification. The combination of good tutorials with online documentation is clearly needed.

4. Knowledge based project management tools, for instance, that would allow a "what if" capability will be required as the information engineering paradigm forces the project manager to consider factors outside of the time frame and technical scope of a single application development. The ability to address resource delays, resource shortages, the addition of tasks, and relation of the time parameters in such a complex domain is essential. The ability to be able to assess local decisions relative to the "current" information system and business strategic plans and critical success factors is required.
5. One of the greatest headaches for a project manager has to do with trying to remember not just what decisions were made, but why they were made at the time. An illustrative scenario involves a meeting of project personnel in which pros and cons for a decision are presented. The manager listens and makes a decision, which is recorded. However the rationale for the decision, and the work that went into producing that rationale, is lost. Much later, the decision is questioned in some different context, and may be reversed for the wrong reasons, and with damaging results, because no one remembers the situation that arose before. The problem leads to the requirement for a project manager's electronic notebook that will allow quick referencing of a project history knowledge base of decision rationale.
6. The knowledge based system building tools currently available, such as ART, KEE, or Knowledge Craft, etc., combined with the Lisp machine environment, provide a natural basis for a prototyping capability, particularly in the model construction support tool arena.
7. Although every information engineering book on the market will assert that the business strategy of a company should drive the information strategy, the process is rarely carried out in practice. In fact, it can be argued that few people know how to carry out that mapping. Consultant tools that aid in delivering the expertise of those few, even by prompting a user to ask the right questions, would be a boon to business directed use of information.
8. The IISSE project is highly unusual with regards to its plan to construct development frameworks for IDS-like implementations, since the usual case is not to plan in anticipatory fashion but to evolve such frameworks out of repeated experience. One IDS prototype has been built. Eventually more systems of the type will follow, and these new system development projects will have the framework results of IISSE available to them. In order to evaluate and improve these development procedures, relevant data must be collected as new implementations occur. Researchers within the IISSE project are people who will know what information needs to be collected, and they will not

likely be party directly to such developments. Thus, one output of the IISSE effort should be a knowledge based system for collecting and analyzing information about the formulation and subsequent usability of specific frameworks.

9. A need that has been uncovered in the current IDS prototyping project at Rockwell involves use of a tool that could analyze a data base for the semantics of a change to the data base. A typical scenario occurs when a change is made to a field in a record. That is an easy change. The hard part is tracing back the implications of the change in terms of the underlying information model semantics. The mapping of a logical to a physical model (and vice versa), plus carrying along the semantics of the transformation is a frequently recurring problem that could perhaps be addressed using knowledge based techniques.
10. Configuration management represents an area which might be supported by knowledge based techniques. Regular checks on items such as identifying files that have been updated are straightforward to implement with traditional techniques, but analyzing the probable implications of irregularities is a knowledge intensive process.
11. Given an environment in which the design artifacts that describe a system are managed in an integrated way, the possibility may exist for generating recommended tests that verify the code against the design specifications.
12. A common theme in recent knowledge based work is one that addresses the reusability notion of software - whether it be reuse of code modules, data structures, or cost estimation. However, in every case that we have noted, the key to reuse is the determination of a "most similar" case history. Considerable research needs to be done in order to gain an understanding of what constitutes "relative similarity" in these cases.

3.4 Technology Transfer Requirements

Technology transfer can be a key issue in determining the success or the failure of a technical program. For example, IISSE concepts and methods must be transferred effectively to a broad-based user company goals. Technology transfer can take place in four separate and somewhat distinct forms. Each of these forms of technology transfer have their own characteristics and requirements in order to be effective. The four widely recognized technology transfer mechanisms are:

1. Direct Technology Transfer,
2. Industry Training,
3. Formal Education, and
4. Publication.

A brief discussion of each of these technology transfer forms is given below to set the stage for a discussion of IISSE technology transfer requirements in a subsequent section.

3.4.1 Direct Technology Transfer

This form of technology transfer is represented by situations in which artifacts are physically transferred to a user. This situation is often encountered when a user is sold or is presented with a software package, a piece of equipment, or a tool; the technology is given to the user. Direct technology transfer is risky and can be very ineffective when the transfer does not involve training since the user is left on his own to interpret how to utilize the technology. It is doubtful that much of the IISSE education and/or training will be done under direct technology transfer.

3.4.2 Industry Training

The training of individuals in IISSE concepts and methods can be effectively achieved with the use of carefully written training manuals, carefully formulated training curricula and training sessions structured for the user. When complemented with up-to-date audio-visual equipment and technology, industry training sessions delivered in an instructor-lecture format can be an extremely effective and inexpensive method of transferring technology. Many user oriented tools and technologies of today are transferred to the user community utilizing this mode of delivery. Industry training emphasizes the "how to" and the mechanics of a procedure or technology and not necessarily answering the question of why we solve a problem in a particular way.

3.4.3 Formal Education

The most popular form of technology transfer is formal classroom training. For an IISSE type of project, formal educational training usually involves the development of a theoretical base and a technical foundation for the method or technology being transferred. In this type of technology transfer, a great deal of emphasis is placed on establishing why a concept, tool, or procedure is being used and how it was developed and justified. Due to the nature of this type of technology transfer, it usually is implemented in a formal classroom environment and is most often offered by educational institutions.

3.4.4 Publications

Through journal articles and technical publications, a concept, practice, or a procedure can be described or transferred to a wide range of readers. In the IISSE environment, there are a limited number of publications that print material that can be classified as educational or technology transfer materials. This form of technology transfer can be effective when certain author-reader combinations are formed. In some instances, authors do not relate the material to be learned by the audience effectively to their readers and the technology transfer effectiveness drops off. The overall technology transfer value of publications is difficult to predict unless a group of readers can be tested and the manuscript revised to reflect suggested changes and to achieve maximum technology transfer.

3.4.5 Technology Transfer Requirements

IISSE development and implementation represents a unique form of systems development. The theories, methodologies, and tools necessary to achieve an IISSE are only now beginning to mature. The fact that there are number of significant managerial and technical skills which must be learned, understood, and applied to an IISSE project is becoming quite obvious. Education and/or training support for the following areas are required for a successful application of the IISSE concepts.

- Technical Skills - An Applications level understanding of the following technical skills required as a minimum for undertaking an IISSE program:
 - Systems engineering methodologies (SEM) and systems development methodologies (SDM) suitably modified to support an IISSE
 - IDEF0 - Function Modeling Methodologies
 - IDEF1 - Information Modeling Methodologies
 - IDEF2 - Dynamics Modeling Methods
 - Data Base Design Methodologies

- Managerial Skills

- Project technical administrative skills
- Project control skills (time and resources)
- Knowledge of the lifecycles (technical and administrative)
- Information Resource Management

We recommend that due to the complexities and uniqueness of IISEE, technology transfer training should consist of three distinct modes:

- Concept Training

- Technical and administrative theories and practices
 - Tools and the principle upon which they are based
 - Methodologies and the procedures for gaining the most benefits from them
 - Methods and the application framework within which they have been developed to operate
- Test Case Analysis of an actual hands-on experience in a test-bed environment - Such an experience should involve a needs analysis, requirements definition, and a prototype application of fundamental concepts using real data and real problems.
 - Context Analysis - A detailed investigation using case study analysis into the strengths and weaknesses of the principles.

It should be pointed out that of these three approaches, only the first has been developed and explored to any large degree. IISEE technology transfer must take into account these three approaches within the structure of each of the four technology transfer mechanisms discussed earlier.

An integrated approach with common technology threads moving throughout each technology transfer mechanism needs to be undertaken. In addition, the technology transfer mechanisms should be formulated to take into account the various levels of management and personnel that will be utilizing the materials. In terms of direct transfer, a guidebook with supporting software should be developed. The software can be developed in a tutorial format for use by all levels within the organization that will be using it. For industry training, slide sets, scripts, and seminar type teaching materials can be developed for the major phases of the IISEE definition and use. To satisfy the needs of the university community, a textbook devoted to the formal teaching of the tools and methods of IISEE needs to be written. Publications spread the technology and educate a far wider audience than any other technology transfer technique. There are several methods for encouraging publication but the most

effective would be to establish a technical journal or newsletter devoted to publishing papers in the area of and IISSE framework. A university is a very good host for such a journal or publication mechanism. The Technology Transfer Thrust provided in Section 3.5 of this report identifies a set of projects which cover these approaches and mechanisms.

3.5 Research Plan

The organization of the ISEE plan follows the standard format for program layout used by the Air Force Wright Aeronautical Laboratories. Figure 3.2 shows the basic elements of such a plan and their relationships. The process of constructing this plan involved review of:

1. The industry needs and requirements presented in the previous sections of this report.
2. The results of inputs from the TAP members.
3. Discussions with the Rockwell IDS analysis and development team experts.
4. Discussions with the IDS Program office.
5. The state of the art in commercially available methods and tools.
6. The literature on emerging research results and the SEI initiative.
7. The research efforts which were a part of this project.

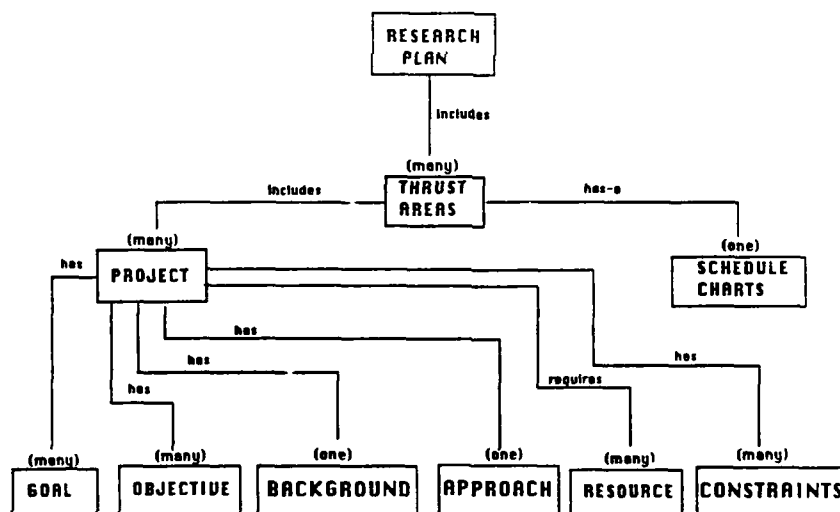


Figure 3.2: ISEE Program Plan Elements

From these inputs we extracted an underlying strategy for providing a comprehensive set of frameworks, methods, tools, and automated environments for supporting an organization in the process of achieving integration through the use of the IDS mechanism.

From the needs and the integration support strategy we designed a methodology and tool development program which consists of eight major technical thrust areas and specifically delineated projects. The design of a tactical plan of this type involves recognizing similarities in the needs, understanding the leverage points and precedence constraints inherent in the realization of solutions to the problems underlying those needs and budgeting out an affordable plan. While we feel that the product which will emerge from the following program has extensive applicability outside the IDS framework we did use the IDS alignment to make the hard decisions necessary to come up with an affordable, achievable program.

The technology thrust areas chosen by the coalition for the plan include:

1. Integration Framework Development
2. Component Methods
3. System Development Environment
4. Component Tools for Management
5. Component Tools for Analysis and Engineering
6. Component Tools for Software Development
7. Knowledge Based Tools
8. Technology Transfer

The following sections describe each thrust area and the associated projects in those areas.

3.5.1 Integration Framework Development Thrust

The projects in this section are focused on the further definition of the IISSE product structure and the construction of the integrating framework component of that structure for several key application environments. The frameworks contained in the plan for this thrust are focused on the development process required to transition an IDS three schema information integration architecture into engineering, manufacturing, and logistics operations. Figure 3.3 displays the current definition of the projects in this thrust area and their relative timing and sequence. This thrust area is also responsible for the overall IISSE program coordination, and the maintenance of external interfaces to other government programs (e.g. IDS, SEI etc).

The following sections provide a description of each of the thrust area projects.

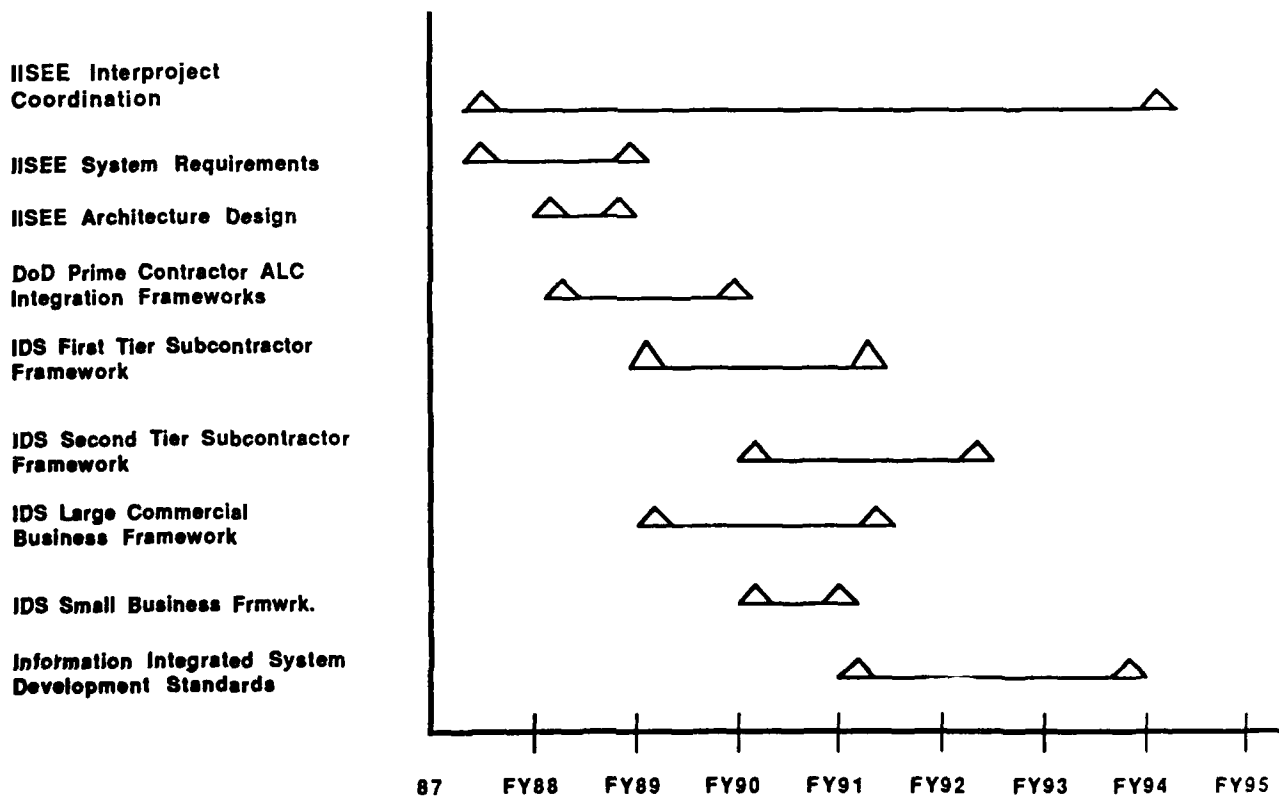


Figure 3.3: Framework Development Thrust

Project Title: IISEE Interproject Coordination

Project Goals: The goal of this project is to provide a co-working / co-development relationship with the IDS development project and the associated IDS implementation projects. The goal of this project also includes the internal IISEE program coordination (inter-project) and technical integration.

Project Objectives: The objectives of this project are to:

1. Provide bi-monthly technical progress briefings to the IDS project office and the IDS technical developers on the results of IISEE program efforts.
2. Participate in the tri-annual TAG meetings.
3. Organize and conduct tri-annual IISEE technical reviews.
4. Analyze the evolving IDS requirements, designs, and problems to identify additional framework, method, tool, or environment needs.
5. Insure that those needs are conveyed to the appropriate IISEE project as action items for resolution.
6. Provide a central point for rapid dissemination of the results of the IISEE program.

Background: The purpose of this project is to provide a coordination and direction setting effort which would span the life of the IISEE program. This project would serve as a channel for methodology and tool needs from the IDS development project to the IISEE program. It would also be tasked to insure that the results of the development projects are quickly assimilated into the IDS development program. This project would be responsible for the configuration management of the IISEE definition and design after the projects which define these items are completed. Thus, for example, this project would maintain / adjust the generic versions of the IDS SDP0 and SDP1 models as well as the generic integration framework method and guidebook as needed.

Approach: The approach suggested for this effort is to establish a continuing assessment and configuration management contract. This program would serve the role as a facilitator (primarily technical) across the life of the IISEE program. Consequently, this project would serve the role of verification and validation on all methods and tools developed as a part of the IISEE program. It would be responsible for enactment of configuration control on the critical components of the IISEE as they are designed. This organization would maintain the library of technical reports and software and would be responsible for the dissemination of those materials. While the ultimate contracting responsibility would obviously rest with the funding government agencies, this integrating contractor would serve to provide a continuity across the life of the program.

Resources: This project would require approximately four man-years per year of technical effort and three man-years worth of administrative support effort per year.

Timing: This project would need to be started immediately and would run the length of the IISSE program.

Constraints: There are no special considerations or constraints on the establishment of this project.

Project Title: IISSE System Requirements

Project Goals: The goal of this project is to complete the definition of the requirements for an Integrated Information System Evolution Environment for IDS implementation support.

Project Objectives: The objectives of this project include:

1. Provide a stable definition of the function, information and technology concepts of the IDS information integration strategy.
2. Completion of the IDS System Implementation Process Function Models (i.e., a generic IDS SDP0).
3. Completion of the IDS System Implementation Process Information Models (i.e., a generic IDS SDP1).
4. Definition of the requirements for an IISSE.
5. Development of a conceptual design for an IISSE.

Besides these technical objectives a strategic objective of this project is to involve key industry, government, and university leaders in the development activities required to define the IISSE. This objective will help insure the rapid assimilation of these techniques into use within their organizations.

Background: In order for the IISSE to be successful, its development must be afforded the same rigorous system development process as the "information systems" which it will eventually support. In this planning project we have established the basic industry need categories and most of the critical requirements; however, there is still a need to define how these industry needs translate into a complete set of method, tool, framework, and computerized support environment. The key focus of the modeling of the system development process is the strategic and tactical planning activities. The reason for this focus is that an integrated information system impacts all of the organizations within a company. The IDS represents a business information strategy and as such must be justified in terms of existing strategies and business plans.

One of the crucial needs for IDS assimilation into a business operation is effective techniques for this assessment and alignment. The models and requirements established in this project will serve as the basis for the design of the IISSE products (frameworks, component

methods, automated tools, and development environments). Another key focus of this modeling activity will be the maintenance and support process since information integration is more a long term policy and way of operation than a specific objective. As indicated by the first objective listed above one of the important needs of industry is for a stable definition of what the IDS information integration strategy is and what the implications of that strategy are for information engineering within an organization.

Approach: The approach to this project consists of the establishment of a coalition of industry system developers, Air Force project engineers, and university researchers to build a composite model of the decisions, actions, and activities required to plan, design, implement, and maintain an IDS based information system. The project team will start with the results of three previous related projects:

1. The 1984 ICAM Systems engineering methodology project (1701).
2. The results of the current planning project.
3. The IDS System Development Methodology project.

The coalition team will be partitioned into four working groups. The first group will focus on the business strategy and planning activities. The second will focus on the technical activities of requirements, design, construction, implementation, and maintenance. The third group will focus on the project management activities. The fourth group will focus on the IDS concept stabilization, working with the IDS development contractor, the IDS program office and the ISO and ANSI working groups.

Resources: The estimated resources for this project are six man years of effort which includes the costs of travel and supplies.

Timing: Due to the amount of review and coordination required to prepare an industry acceptable requirements statement, it is recommended that this project be scheduled for a minimum of 16 months.

Constraints: All of the requisite data for this task is available it should start immediately.

Project Title: IISEE Architecture Design

Project Goals: The goal of this project is to define the IISEE architecture which would delineate the IISEE components (as illustrated in Section 2.4 of this report), develop the philosophy of IISEE operation and use, and integrate of the IISEE with the ongoing IDS development efforts.

Project Objectives: The objective of this project is to take the conceptual design of the IISEE and expand it into detailed specifications for each component. The primary focus of this design effort will be the interface specifications which describe how the pieces

of this design will be integrated together. This project would also be responsible for the establishment of the generic framework structure and guidebook components of the IISEE.

Background: The IISEE program must address several near term framework, method, and tool needs. However, the key requirement for an IISEE is for the integration of all of these needs into a unified life cycle package. The detailed design on the IISEE will be focused on making the IISEE itself integrated. This effort entails the clear definition of all of the interfaces between the actions specified in the framework and insuring that the methods produce the data needed to support not only the activities themselves but also the decisions needed to go from one activity to another. This detailing also requires a complete human factors analysis of the tools designed to support the framework and methods. The user interface of the tools must be structured to support the human decision process specified by the framework at the point in time of the tool application.

Approach: The approach recommended for the detailed design project is to use what has been referred to as a scenario driven approach. This approach starts with a conceptual design of the IISEE and (using walk through by experts) identifies the detailed specification of the form and function of the required interfaces. Such an approach requires the organization of a design team with a chief designer and a coalition of contributing experts. Central to this team is the IDS developers who represent the leading experts in IDS style integration. However, there are other organizations who have implemented similar three schema architecture systems (e.g., the Boeing IISS team) who would have to be identified and assembled as a part of the design team. Another critical aspect of the design of the IISEE is how well the automated support tools will integrate with the IDS system. Since IDS is still under active development and expansion, a forum must be established for the IISEE design team to submit proposed integration mechanism designs to the IDS design configuration management organization for disposition.

Resources: The resources required for this task are estimated at five manyears for development and three manyears for review.

Timing: This task is estimated as a nine to twelve month activity.

Constraints: The principle constraint on the initiation of this task is the completion of the model portions of the requirements project described above.

Project Title: DoD Prime Contractor and ALC Integration Frameworks

Project Goals: The goal of this project is to develop the IISEE framework component to be used by DoD prime contractors for strategic and tactical planning, design, and implementation of an IDS based engineering information integration system. This framework represents the integration method often referred to in the company needs and requirements section of this report. But "integration" is not achieved through the application of a single method to a single project. Integration is achieved by the focused application of a number

of methods towards a common, well defined goal. The frameworks described in the following projects provide the structure for application of this discipline across a number of projects within the respective environments which will conclude in an integration directed program within the organization.

Project Objectives: The objective of this project includes the detailed design, development, and documentation of the IISSE framework component described above. A second objective of this project would be the prototype implementation of IDS in two prime contractor facilities and one ALC for the purpose of bootstrapping the development of the frameworks in actual application settings. This boot strapping activity would use the initial results from the IDS development experience and the initial tools and methods from the parallel development efforts in other thrusts.

Background: The first framework to be developed will address the needs and context of application of the IDS framework in a large defense contractor environment. This focus should allow for the most direct transfer of the Rockwell experience and will also cover the application to large AF ALCs.

Approach: A framework is primarily a methodization of best practice. To be successful, it must prescribe actions and decisions which the using organization can carry out. Therefore it represents the capture of the experience base and lessons learned in one organization in a form usable by another. Because it is difficult to clearly define "what" the critical knowledge of the source organization is and what the recipient organization needs, the approach recommended involves the establishment of an "apprentice" program. Through this program, two prime contractor organizations and one Air Force logistics center would be set up as apprentices to the IDS development project with a third organization acting as a facilitator and recorder. The apprenticeship program would last through the implementation of a prototype IDS within the sponsored organization's companies.

Resources: Since the approach recommended involves the development of two IDS implementations this project is expected to require on the order of 30 manyears of effort (although cost sharing from the individual companies could significantly reduce this figure.)

Timing: This project is estimated to require 18 to 24 months to complete.

Constraints: The primary constraint on the performance of this task is the availability of the IDS development team.

Project Title: IDS First Tier Subcontractor Framework

Project Goals: The goal of this project is to develop the IISSE framework component to be used by DoD first tier subcontractors for strategic and tactical planning, design, and implementation of an IDS based engineering information integration system.

Project Objectives: The objective of this project includes the detailed design, development, and documentation of the IISSE framework component described above for application

to first tier subcontractor organizations. This project would also result in three prototype IDS implementations in first tier subcontractor facilities.

Background: The first tier subcontractor situation is similar in scope to the prime contractor problem but generally different in organizational impact. That is, the first tier subcontractor generally has as complex an environment as the prime contractor in that the environment generally represents a single division in the prime while it represents the entire company of the first tier subcontractor. Thus the implementation of an IDS has much more extensive impact on the total organization.

Approach: It has been shown that one of the most effective ways to perform technology transfer from the primes to the subcontractors is through the primes themselves. Therefore the approach suggested is to match each of the primes who have participated in the previous contract with one of their subcontractors and a facilitator to repeat the process described for the prime of implementing the IDS in the subcontractor environment. The IDS developers would be involved in each team as supporting consultants.

Resources: The estimated resources for this effort is 25 manyears.

Timing: This project is estimated to require 24 to 30 months.

Constraints: This project could start as early as the detailed design phase of the prime contractor framework development project.

Project Title: IDS Second Tier Subcontractor Framework

Project Goals: The goal of this project is to develop the IISEE framework component to be used by DoD second tier subcontractors for strategic and tactical planning, design, and implementation of an IDS based engineering information integration system.

Project Objectives: The objective of this project includes the detailed design, development, and documentation of the IISEE framework component described above for application to first tier subcontractor organizations. This project will also result in three prototype implementations of IDS in second tier subcontractor facilities.

Background: The second tier subcontractor problem of IDS implementation is similar to the first tier in impact on the organization but is expected to be of smaller implementation scale.

Approach: The approach recommended for this project is identical in concept to that proposed for the development of the first tier subcontractor framework development. A set of three teams would be established. Each team would consist of a first tier subcontractor who had participated in the first tier subcontractor integration framework development and one of his subcontractors. The prime contractor would be involved in a support consulting role and, as before, a facilitator would be provided on each team to record the development methods as they are defined.

Resources: The estimated resources for this effort is 20 manyears.

Timing: This project is estimated to require 24 to 30 months.

Constraints: This project could start as early as the detailed design phase of the first tier subcontractor framework development project.

Project Title: IDS Large Commercial Business Framework

Project Goals: The goal of this project is to develop the IISSE framework component to be used by large commercial organizations for strategic and tactical planning, design, and implementation of an IDS based engineering information integration system. This development is important to insure that the commercial and military sides of the organization have a common pathway to integration and control of their product data.

Project Objectives: The objective of this project is to establish an integration framework effective for application in large commercial business environments and to demonstrate a prototype IDS based information integration mechanism within such an organization.

Background: The ultimate success of the IDS concept will be measured by its acceptance at large both within the military and commercial business sectors. It is only with wide spread acceptance and support of the concept that the economies of scale will drive the technology to an acceptable cost and risk position. This project is aimed directly at acceleration of the IDS concept by industry at large by providing a customized framework for commercial use. This initial framework will be targeted towards the large scale commercial enterprise both because of similarity to the prime contractor framework and because the needs analysis has indicated that the business sector is keenly aware of the needs and has the internal talent and resources to take advantage of the solution.

Approach: The approach recommended for this project is to team the facilitator, which has been involved in the previously defined integration framework developments, with the commercial company for the development of the framework. The IDS contractor would be involved in a consulting support role. The process would parallel that used in the previously described framework development projects. The benefit to the company would be the assistance and partial offset of costs required to implement a prototype information control system. The cost would be that the framework developed would be turned into the public domain.

Resources: Because of the investment sharing anticipated with the approach outlined above, it is anticipated that the overall cost of this project will be limited to 8 manyears.

Timing: This project is expected to require 16 months.

Constraints: This project can be started concurrently with the first tier subcontractor framework project.

Project Title: IDS Small Business Framework

Project Goals: The goal of this project is to develop the IISEE framework component to be used by small business (private or subtier DoD contractors) for strategic and tactical planning, design, and implementation of an IDS based engineering information integration system.

Project Objectives: The objective of this project is to establish an integration framework effective for application in small business environments and to demonstrate a prototype IDS based information integration mechanism within such an organization.

Background: Simply considering the large percentage of U.S. industry which is small business and the percentage of those companies who produce (or will produce) DoD goods and services justifies the tailoring of an IDS and IISEE framework to service this community. One advantage of the small business application arena is the relative lack of legacy systems and the aggressive decision making style which is prevalent in this industry sector. These factors should allow for a streamlining of the IISEE framework rather than an elaboration. The benefits to this sector in terms of government contracting potential combined with the inherent increases in competitive advantage should generate strong interest in the technology if it can be put in a form accessible to these organizations.

Approach: The approach to this project would require careful selection of at least two small businesses with high visibility in the national community and sophisticated enough operations that the IDS concepts have merit in their environments. These participants would be paired with one of the participants in the first or second tier subcontractor framework developments and a facilitator. The resulting initiatives would have to work closely with the IDS developers to determine how to reduce the scale, complexity and resource requirements for application in their level of facilities.

Resources: The resources required for this effort are estimated at 6 manyears.

Timing: The time period required for this effort is estimated at 12 to 16 months.

Constraints: The first tier subcontractor project should be completed prior to the initialization of this effort.

Project Title: Information Integrated System Development Standards

Project Goals: The goal of this project is to provide a focus and organizational resource to support industry initiatives in the development of standards for the application of the IISEE. The intent of this project would be to accelerate the identification of areas for profitable development of standards and the recognition of these areas by various standards making bodies (e.g., NBS, Ansi, Codasyl, and DoD agencies).

Project Objectives: The objective of this project is to establish an information integrated systems development standards group and to take the necessary steps to inject this area into the official standards making organizations.

Background: In the long range implications of IDS technology on the military industrial complex, some level of standardization between the product data control models (PDCMs) for those portions of the critical life cycle product data is needed. The purpose of this standards making activity is to identify the lowest level of standardization for these structures which will allow the desired data communications between these IDS centers.

Approach: The approach to the establishment of such a standards directed activity should be formulated with the assistance of representatives from the Codasyl Systems Committee, the PEDES organization, and the Air Force.

Resources: A rough estimate of the resources for this activity would be three manyears per year for coordination and support. It would be anticipated that most of the costs for the participating members would be absorbed by the organizations in the process of implementing IDS systems.

Timing: This project is estimated to require three years to develop a stable working group, identify the major areas for standards considerations, and start the transition to an established standards making group.

Constraints: This project should not start until the initial frameworks have been established and the IDS prototype implementations are underway.

3.5.2 Component Methods Development Thrust

The projects in this thrust area represent the critical component methods of the ISEE which need to be developed. Figure 3.4 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation. This thrust contains an emphasis on the formalization of all methods to be used in the ISEE to insure integratability of the methods and to form a basis for evolving a method engineering discipline.

The following sections provide a description of each of the thrust area projects.

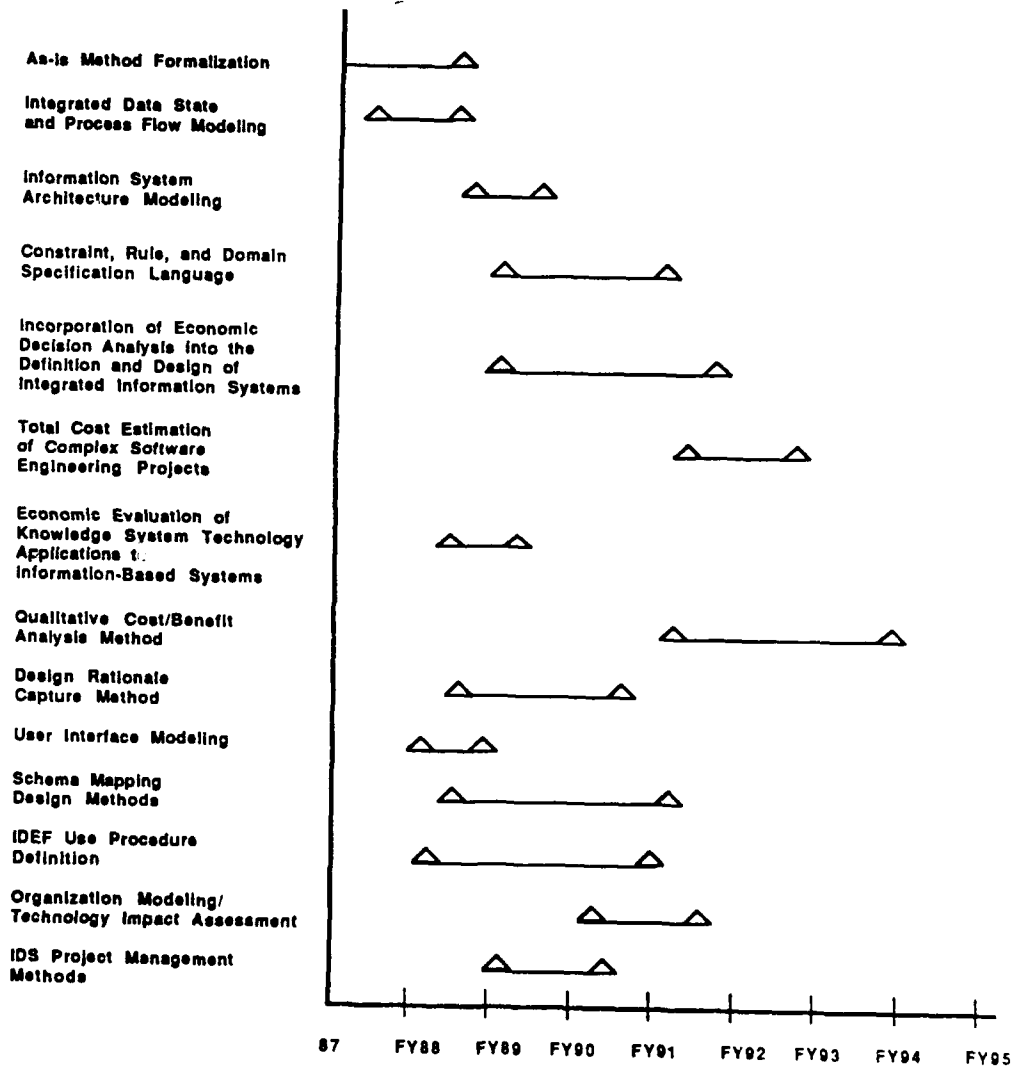


Figure 3.4: Component Method Development Thrust

AD-A195 051

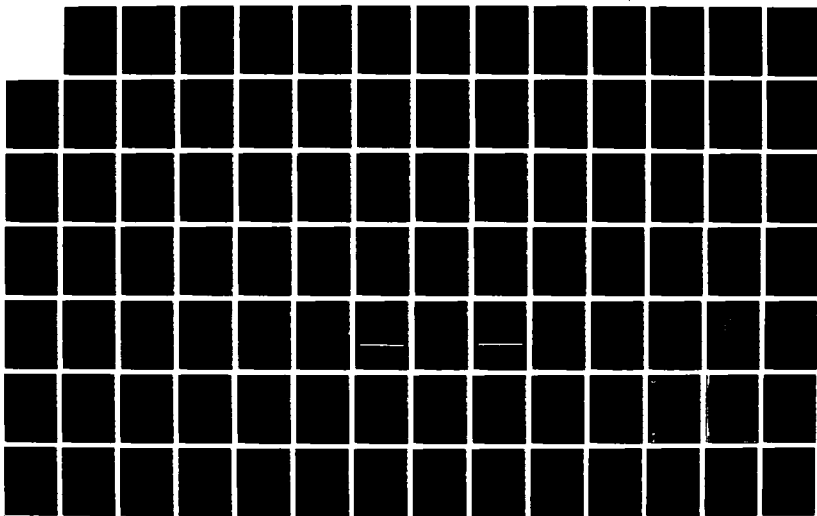
KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS
DEVELOPMENT METHODOLOGIES. (U) MASSACHUSETTS INST OF
TECH CAMBRIDGE A GUPTA ET AL. DEC 87 HIT-KBIISE-2
DTR557-85-C-00003

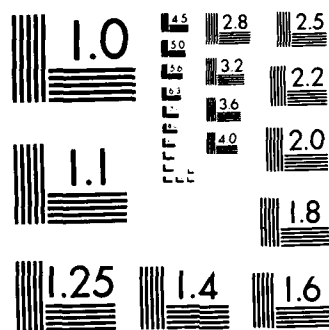
2/4

UNCLASSIFIED

F/B 12/7

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Project Title: As-Is Method Formalization

Project Goals: The goal of this project is to provide a rigorous formal analysis and comparisons of the major existing system planning, analysis, and engineering modeling techniques, and to provide a sound basis on which to fill perceived modeling voids with the extension of these techniques and development of new techniques.

Project Objectives: Phase I of this project will develop the syntactic and semantic formalization of the following information and data modeling techniques: IDEF1-X, NIAM (ENALIM), NIAM (Data Model) and ER. Phase I will also establish a standard for the formalization of all future methods developed under the IISSE program. Phase II will focus on the formalization of IDEF0 function and NIAM process modeling techniques. Phase III will focus on the formalization of database and application program design support modeling techniques.

Background: Effective analysis, design, and maintenance of large-scale integrated information systems requires the use of flexible, clearly and unambiguously defined information system modeling techniques to support planning, analysis and design activities. Nearly all currently existing techniques lack the sort of rigorous formal basis that will permit the integrated, flexible usage that is required. More exactly, there is no mathematically precise formalization of these techniques that will yield precise answers regarding their consistency, relative expressive power, and their intended meanings. This deficiency also severely hampers the teachability of the techniques and limits a trainer's capacity to test the competence of the trainees. Recent developments in the study of formal syntax and semantics have provided all the tools that are needed for the sort formal basis envisioned here; indeed, this task has already to a large extent been carried out in the work on IDEF1 reported in Chapter 4 of this report. This project would consist of the further application of these tools to the syntactic and semantic formalization of the remaining major information system modeling techniques. After the individual formalizations are complete, work will then begin on formal comparisons of the various techniques, extensions of those techniques, and building on the insights gained from this work, the development of new techniques that are more general, more expressive, and more flexible still.

Approach: The formalization process is a combination of the following theoretical and empirical activities, performed roughly in sequence:

- Examine existing documentation for the technique in question and begin extracting its underlying syntax and intended semantics.
- Attempt to formalize the syntax and semantics. This involves the choice of a lexicon, the development of formal grammatical rules, and the design of an appropriate mathematical interpretation of the syntax in terms of functions, relations, sets, and other mathematical objects.
- Identify problems within the formalization (e.g., possible misinterpretations of the technique), as well as apparent problems uncovered within the technique being formalized

(e.g., identification of redundant or inconsistent concepts in the technique).

- Work closely with the original developers to clear up misinterpretations, to introduce them to the formalization as far as it has been developed, and to gain their assistance and guidance in continuing the formal development.
- Show that each well-formed syntactical "model" generated by the grammar has an interpretation, i.e., a set theoretic representation of a possible real-world modeling situation. This ensures that the grammatical rules are sound.
- Show that each interpretation can be represented in the syntax. This ensures that the grammar can represent any possible modeling situation which it is designed to capture.
- Suggest possible extensions to the modeling technique in question.
- Begin work on formal comparisons with previously formalized modeling techniques.
- Use insights gained to continue work on the more general modeling technique.

Resources: The resources required for this project can be estimated at about one man year per method. This includes time and travel costs of the formalization experts, the method developer, and at least one expert method practitioner.

Timing: Because of the basis already established in previous projects, this project can start immediately. The total time span for this project is estimated at 18 months.

Constraints: This project will provide the technical basis for the development of the Neutral Information Representation Scheme and for many of the component tool developments. Therefore, it should be coordinated with those activities to provide the proper technical data to the respective development teams.

Project Title: Integrated Data State and Process Flow Modeling

Project Goals: The goal of this project is to fill two critical method voids (data state and process flow modeling) identified in the course of this study as limiting the capability to implement three schema systems, such as the IDS. The design goal of this project is to design these new methods from the formalisms developed for the IDEF methods so that they represent logical extensions which are integrated in concept and practice.

Project Objectives: The objectives of this project include the definition, design, and development of the concepts, theory, syntax, procedures, and usage for data state modeling and process flow modeling in engineering information systems. The deliverables from this project will be a three volume set of technical documentation for each method covering

respectively the definition, discipline, and use of the method. This project will also define the automation support requirements for these methods.

Background: The product definition data evolves from the original requirements in the mission definition through various levels of engineering maturity, through manufacturing, operations, and logistics support. Current information modeling methods are limited to the capture of a time independent view of the composition and relations of information. While this is a critical view of the information, to produce a product data management and control system which spans the data life cycle it is necessary to clearly map out the states through which the data proceeds. Along with the state definition should be included the representation of the logic for state transition (e.g., the organizational mechanisms, the preconditions, and the resulting actions associated with a transition). The key to the design of such a method is the design of a simple approach to the acquisition, display, management, and use of this complex set of information.

The current IDEF0 provides an excellent mechanism for display of a hierarchical representation of the activities in an organization. However, for the documentation of the flow of major activities associated with the product data life cycle, a process rather than functional view is required. The process view would display the data states (rather than data objects as in IDEF0) which are preconditions on the initiation of an operation or process and the changes in those states caused by the process.

The reason for developing these two capabilities together is to insure that the resulting two methods are integrated both in concept and application.

Approach: The approach recommended for this project involves the use of both the experience base of experienced systems developers as well as some of the previous research work in the area of data design specification languages. The following steps encompass the basic approach which would be required for each method:

1. Refine the requirements for the data state and process flow methods.
2. Collect sample specifications of this information in the form currently used by practicing system developers.
3. Review the research work in related methods including DDS, IDEF1/ES, Petrie Nets, VHDL, ADLIB, GMB, and CCS.
4. Work with the model formalization project to formulate the basic concepts and theory of a data state and process flow method consistent with existing function and information modeling techniques.
5. Design a graphical and language syntax for the display of the information to be acquired using the methods.
6. Design appropriate forms and methods for the acquisition of the required information to populate the models.

7. Coordinate the design activities with the component tool development thrust.
8. Apply the developed methods to a test case situation and revise / refine the components.
9. Submit the developed methods for review by a group of information system developers (including current IDS developers).

Resources: Previous experience with method development has shown that approximately 2 manyears per method is required for the basic development. Another 1 manyear is required to develop the first generation management orientation and technical training materials. To support the testing of the methods in an actual situation, at least 1.5 manyears per method should be planned.

Timing: This project will require at least 18 months to complete with the initial method developments being completed within the first 12 of those months

Constraints: This work could start within 6 months of the formalization project.

Project Title: Information System Architecture Modeling

Project Goals: The goal of this project is to develop the method and performance/cost analysis techniques to allow the modeling of both structural and physical aspects of information system architectures. These techniques are required in order to insure that the requirements of the strategic information plans and logical system designs will be realizable with available hardware, software, database, and communication technology.

Project Objectives: The objectives for this project include:

1. Development of a capability to represent information system structural and physical architecture (both graphically and textually).
2. Development of a method for mapping between the logical information system architectures and this implementation view.
3. Development of a method for management of requirement impact based on changes to the implementation architecture.
4. Development of the simulation and analytic modeling techniques for performance prediction of the implementation architectures.

Background: The rapid change in computer and communications technology demands a sophisticated mechanism for easily representing an implementation architecture and mapping the logical system design onto that architecture. Several systems such as AISIM have

been developed for the Air Force which allow the simulation of complex computer and communication systems from a model library which can be easily extended to accommodate new technology. Such research products can be used as guidelines for the definition of what representation symbology and language would be effective in the design of this capability.

Resources: This project is estimated to require three man years of effort.

Timing: The time frame for development of the definition, discipline, and use components of this method is estimated at 12 months.

Constraints: It is recommended that the data state and process flow modeling techniques be structured prior to the development of this capability.

Project Title: Constraint, Rule, and Domain Specification Language

Project Goals: The goal of this project is to develop a standard language for relation constraint specifications, business rule representation, and attribute value domain specification which could be used by a wide variety of information modeling methods.

Project Objectives: The objectives of this project include:

1. Definition of the language requirements for the three languages identified above.
2. Design of a syntax and semantics for each language which takes into account ease of use for specification, ease of interpretation, and ease of computerization.
3. Experimentation with the application of the languages in a practical setting to allow tuning of the language to the needs and capabilities of the target user population.

Background: None of the existing information modeling methods have a complete complement of:

1. Relation constraint specification languages for both procedural and declarative constraints. (e.g. roles, assertions, timing, and sequencing or general timing constraints such as the currency of fact relative to event which produced fact)
2. Business / engineering / manufacturing operations rule representation.
3. Attribute value domain specification.

What is needed is a coordinated approach to the development of these languages consistent with the formal theory of as large a representation of the current information modeling methods. The key to a successful development of the required family of languages is maintaining a careful balance of efficiency, expressiveness, and formalism in the definition of these languages. Another requirement on the language development is that the resulting languages be usable with (extensions of) existing methods.

Approach: One fact which we know from the past 30 years of programming and specification language design is that the requirements of a language must be defined by the user community but the design of the language should be performed and controlled by a small, dedicated team. This is exactly the approach which is recommended. It is also recommended that the approach attempt to build on concepts in existing languages (such as the NIAM RIDLE, OBJ, and IDEF1/ES rule languages). Finally, it is recommended that the language development process be paired with application of the language in the framework development projects so that the concepts and structure can be tested and evolved with actual field use.

Resources: The resources required for this project are estimated to be on the order of six man years.

Timing: The development of the required languages and the testing needed to ensure usability is estimated to take 18 to 24 months.

Constraints: This project could start as soon as the initial work on the data state and process flow methods is complete.

Project Title: Incorporation of Economic Decision Analysis into the Definition and Design of Integrated Information Systems

Project Goal: The goal of the project is to specifically incorporate the modeling and analysis techniques of information economics into the needs analysis and design phases of large information systems which require integrated systems development.

Project Objectives: The specific objective of this project is to establish the methods necessary to identify the payback potential of information integration programs within the following target business types:

1. DoD prime contractors
2. DoD first tier subcontractors
3. DoD second tier subcontractors
4. Air Force Logistics centers
5. Large commercial organizations
6. Small business

Background: Much of the integration that occurs in the science of information development and utilization requires that existing systems be linked through interfaces, some of which have knowledge systems or expert systems as their implementing medium. The remainder of integrated information systems involve the design and development of new, broad-based

systems with complex databases (new and existing) in several functions of the enterprise. Commonly, the requirements for data collection, storage, manipulation, and retrieval are not considered during the design phase of the projects. Thus, the economic aspects of information systems are neglected, and the users inherit expensive to operate and maintain systems. This project would concentrate upon the development of an automated evaluation system for data collection, manipulation, storage, retrieval, etc., to be utilized during integrated information system design. The deliverable would be a software system used by information system designers as a guide to data item selection and by data system personnel in the design of the data collection procedures for items to be included in large information systems.

Approach: The approach recommended for this project involves a two phase pass through the development life cycle. The first pass will produce functional methods and experimental software which can be used by the IDS implementation efforts associated with the framework development projects. The second pass will build on the experiences of these users and develop distributable methods and software.

Resources: This project is estimated to require four manyears of effort for Phase I and five manyears for Phase II.

Timing: The time frame for completion of Phase I is estimated at 16 months. The duration of Phase II would be determined by the results of the use trials.

Constraints: None.

Project Title: Total Cost Estimation of Complex Software Engineering Projects

Project Goal: The goal of this project is to provide improved techniques for the estimation of costs in the development of large information integrated engineering and manufacturing systems.

Project Objectives: The project objective is to prepare, test, and prototype in an engineering and manufacturing environment a comprehensive cost estimation methodology and procedure for large, complex software development projects.

Background: Much of the cost for software design, development, test, etc., is experienced in support functions such as management, planning, information system integration, resource allocation, and maintenance. The use of the unified life cycle as a basis for software development cost estimation can lead to the development of a comprehensive methodology for software planning and development. Incorporation of the currently available technical tools and procedures for actual code development (COCOMO, etc.) with the efforts necessary for the management aspects of large software projects will give a much more realistic total system cost estimate. A methodology recently developed indicates that its enhancement and prototyping in a manufacturing environment would reduce some of the large software cost overruns being experienced.

Approach: The approach recommended for this project recognizes the fact that as a pro-

gram evolves, the cost drivers and benefit categories become better understood. Therefore, what is recommended is the development of a set of cost estimation techniques with different characteristics to be used at different points in the life cycle of an individual application. To make such a scheme work requires the methods developed to have the capability to identify not only the estimated costs but also the decisions / technology factors which will potentially cause variations and the anticipated magnitude of these variations. The method set must be integrated to the point of allowing the development of the detailed cost / benefit estimations at later phases of the application development life cycle off of the results of the previous estimates with the additional information.

The approach taken for this method development must also recognize the difference in cost estimation for application development and cost estimation for an information integration program. A separate set of methods should be investigated for the latter, based on techniques similar to those used to estimate the cost and benefits to other major programs such as a quality improvement program.

Resources: The estimated resource requirement of this project is five man years.

Timing: The estimated time required for this project is 16 months.

Constraints: The development of the first set of methods could proceed immediately, but the second set will require at least the first set of IDS integration framework developments to be complete.

Project Title: Economic Evaluation of Knowledge System Technology Applications to Information-Based Systems

Project Goal: The goal of this project is to design and develop an economy based evaluation system which can specifically address the technology of knowledge based systems when this technology is being considered as a means to materially increase the performance capabilities of a currently operating engineering system or function.

Project Objectives: The objective of this project is to develop and demonstrate a method for assessing the cost benefits of knowledge based systems to engineering applications.

Background: All too often, the knowledge system, especially in the form of an expert system, is viewed as the natural and most cost effective means to improve the performance of an existing system or to capture the knowledge present in an aging, but essential, system or group of people. Often, however, the implemented knowledge system is found to be a disappointment because of its poor acceptance and utilization, incompleteness, or poor cost performance. This project would involve the incorporation of currently available cost and evaluation methods, techniques, and modeling systems for use by a system manager. Evaluation of the economic parameters of the potential knowledge based system would be accomplished through use in such a fashion that pertinent cost factors would be defined, estimated, and evaluated. This would force the decision maker to carefully examine the

costs and benefits of the anticipated system during the feasibility study (when needs and requirements are defined and examined), not once the design is in process or completed.

Resources: The resource estimate for this project is estimated at two man years.

Timing: The time requirements for this project is ten months.

Project Title: Qualitative Cost / Benefit Analysis Method

Project Goal: The goal of this project is to structure the use of qualitative benefit assessment, analysis, and comparison techniques for the non-tangible factors in technology evaluation.

Project Objectives: The project objective is to develop and prototype an analysis process which incorporates and quantifies factors which are usually considered intangibles with the economic productivity factor (described above) for use in making technology based decisions in information, automation, etc.

Background: The concept of system value is commonly employed by the system designer when technology decisions must be made, but quantification of the intangible factors is seldom attempted. Rather, the economic aspects of the system project are detailed and the remaining intangible factors, regardless of their importance, are listed in some form and not specifically considered an integral part of the evaluation process. There are at least four major attributes to be considered when a technology improvement decision is necessary:

1. Correspondence with strategic plan
2. Design capability
3. System performance
4. Economic performance

There are several factors which can be defined and tailored for each of these attributes, and the importance of each factor can be determined and quantified through techniques such as the Analytic Hierarchy Process (or others). The deliverable of such a project is the process, weighting system, and technique tailored to technology, especially integrated information systems, and a prototype implementation of the complete methodology. Current development of the analysis process for a manufacturing environment indicates that such an approach can rapidly put into perspective the importance of economic versus performance versus strategic technology issues when decisions vital to the future technology level of an enterprise are being made.

Approach: This project would be structured as a two phase effort with the first phase focussed on the identification of commonly accepted qualitative factors in technology evaluation. The first phase would also identify the reasoning methods used by human decision

makers in the determination of the relative merits of these factors. The second phase would focus on the methodization of the factor identification and analysis / assessment techniques.

Resources: The estimated resources for this project is three man years for phase I, including coalition support, and two man years for phase II.

Timing: Phase I of this project would require at least twelve months. Phase II could be completed in nine months.

Constraints: This project should be coordinated with the previously identified cost estimation method development projects.

Project Title: Design Rationale Capture Method

Project Goals: The goal of this project is to establish the methods needed for supporting the evolution of an IDS implementation through the entire life of the organization.

Project Objectives: The objective for this project is to develop the representational capability to capture information system design rationale and associate that rationale with the design models and documentation for the system.

Background: One of the key problems in the use of archived design data is the lack of any rationale for the structure function relations which exist in the design. This method void is possibly the largest single contributing factor to high maintenance costs, and the glaciation of legacy systems. Even with data dictionaries and complete design structure definitions the maintainer of a system must still guess why the particular structure was chosen or why a particular implementation strategy was taken. The capture and transfer of this design rationale is largely by word of mouth today. This causes inefficient use of the time of key personnel and can cause serious delays or even project failures when team members leave. Unfortunately the capture of this information is not a trivial task. A reasonably sized system requires thousands of individual design and implementation decisions. The rationale for each decision may involve extensive knowledge of the development context at the time of the decision. Some of those decisions are "obvious" based on intimate knowledge of the application environment and some are "obvious" based upon having a "working knowledge" of the technology being applied. Thus, without a well thought out method and a set of tools to support that method, the designer is faced with a situation in which he must attempt to describe a complex design context which may seem obvious. The result is that the really valuable information relating to the assumptions and tradeoffs which constitute the actual engineering activity products is lost. This issue becomes critical in considering an information integrated system program such as an IDS implementation. By definition, these systems will extend through out the life of the corporation. They will have to be continuously evolved in place. Without such a method, the only means of accomplishing this task is the establishment of a large organization to maintain the system.

Approach: The approach to this project includes the following tasks:

1. Apply knowledge engineering methods to the study of how information system designers rationalize design decisions.
2. Formalize the concepts identified in the study phase into an information engineering rationale annotation method.
3. Develop a symbol set and grammar for capturing this rationale.
4. Develop a mapping technique for establishing the relation of this information to the design structure and specification methods.
5. Validate the capture and mapping methods through application to the IDS development project.

A two phase project is suggested, with the first phase dedicated to the analysis and formalization efforts and the second phase dedicated to the methodization and validation activities.

Resources: The estimated resource requirement for this project is 15 man years.

Timing: Phase I of this project would require 12 man months. Phase II would require an additional 9 months.

Constraints: This project must be closely coordinated with the knowledge based tool for design rationale capture and the other method developments.

Project Title: User Interface Modeling

Project Goals: The goal of this project is to develop methods for definition of user interface requirements modeling and presentation style user interface design.

Project Objectives: The objective of this project is to:

1. Develop a method for the acquisition of user interaction requirements.
2. Develop a design method for the definition of the interaction scenario, external schema, and presentation layout to support the interaction requirements.
3. Develop a means of mapping these designs to the IDS supported user interface development utilities.

Background: Modeling of the user and how he will interact with a system is a complex problem which is just beginning to be understood. A well designed user interface (UI) should display the following characteristics indicative of the information to be captured in a UI requirements model:

1. Consistency in design, including:

- (a) Message system
- (b) Layout of screen design
- (c) System procedures
- (d) Input and output methods
- (e) Terminology

2. Simplicity and ease of use, including:

- (a) Menu / Form based
- (b) Window-oriented
- (c) Online help
- (d) User's guide
- (e) Tutorial to guide the user through system

Resources: The resources estimated for the development of this method are three man years.

Timing: The time required for this project is one year.

Constraints: This project has no constraining requirements and may begin immediately.

Project Title: Schema Mapping Methods

Project Goals: The goal of this project is to reduce the time and resources required to build the mapping information needed in the IDS implementation architecture.

Project Objectives: The objective of this project is to develop a set of methods to support the production of optimized logic for the mapping between external and conceptual schemas as well as between conceptual and internal schemas.

Background: In order to make the IDS concept work within the definition of the internal, conceptual, and external schemas, one must also define the logical mapping between these views of the data. As this mapping also includes the definition of (and enforcement of) the logical and physical constraints of the information system implementation, development of such mappings is a complex task which can introduce errors into the system.

Approach: The approach recommended for this project is to investigate automatic programming research results (particularly the compiler generator efforts) and the constraint / rule specifications languages being developed in this thrust as a basis for the development of methods for the mapping specification method.

Resources: The estimated resources for this project are 7 man years.

Timing: This project will require 16 to 20 months for completion.

Project Title: IDEF Use Procedure Definition

Project Goals: The goal of this project is to improve the effectiveness of the existing IDEF methods in the IDS implementation efforts.

Project Objectives: The objective of this project is to define a comprehensive set of use procedures for the IDEF methods which would cover the application of these methods throughout the life cycle of an IDS implementation.

Background: The models which result from the application of IDEF methods have been used to support:

1. Strategic planning decisions
2. Scoping decisions
3. Cost analysis and benefits estimation
4. Information system design analysis
5. Database design definition
6. Testing and validation

as well as many other activities within the system development process. Unfortunately no description exists relative to "how" to use the models to accomplish these uses. Therefore, new practitioners must often start from scratch and rediscover for themselves the procedures. This fact was identified as one of the major shortfalls of the IDEF methods by industry. The objective of this project is to capture the successful use of IDEF models and develop procedures to guide future users of the methods.

Approach: The approach to this effort must start with the IISSE, IDS SDP0, and IDS SDP1 models to identify what decisions are supported by the IDEF methods in an IDS implementation. From this structure and the assistance of the IDS development team, the strategies and procedures in the use of the model results can be analyzed and documented. This approach is iterative and should extend over the life of the IDS integrating framework development projects.

Resources: This project will require three man years of effort.

Timing: This project will extend over the lifetime of the integrating framework development projects.

Project Title: Organization Modeling / Technology Impact Assessment

Project Goals: The goal of this project is to improve the capability to assess the current structure and design of an organization and its interaction with the legacy information systems to allow the determination of the impact of an IDS implementation in that environment.

The purpose of this project is to develop a method to augment the tactical planning activity of defining a workable IDS insertion strategy with the associated technology assimilation mechanisms.

Project Objectives: The objective of this project is to define a set of methods for acquisition and representation of the policies, procedures, authorization, decision levels, and structure of an organization and the relationship between these facets of the organization and the information system for the purpose of information strategic planning and the assessment of the potential impact of new technology on the enterprise. The objective of these methods is to provide the data needed to determine:

1. Is the IDS capability desirable?
2. What are the organizational/technological ramifications?
3. How should it be customized to fit?
4. In what order should the capabilities be invoiced?
5. What is the best process of insertion?
6. How should the impact / rate of resultant change be controlled?
7. How to maximize the leverage potential of the new technology?

Background: Implementation of IDS technology or any information integration mechanism within an organization will require (and result in) changes to the organization and procedural systems of that environment. What is needed is a method which allows the determination of what the organizational impact of information integration will be and helps in both redesign of the organization and transition planning. This method should build on the existing IDE methods.

Approach: The approach suggested for this method development is a categorization approach which would proceed with the identification of a frame of reference and then the population of this frame of reference with industry experience. For example, one axis of the frame of reference might contain a variation of Nolan's stages of information system evolution. Another axis of the frame of reference then might include observable facets of an organization (such as management style, technology architecture, etc.) In the cells within this matrix would be observed industry uniformities which would be collected by actual study of a set of industries.

Resources: The estimated resources for this project is five man years in a team framework. Two of these man years are for the development of the framework itself. The other three is for the population of the anticipated features.

Timing: This project will require nine to twelve months to develop and validate the described methods.

Project Title: IDS Project Management Methods

Project Goals: The goal of this project is to modify / extend traditional application software project management methods to adapt to the complexity and longevity of an IDS implementation within an organization.

Project Objectives: The objectives of this effort are to develop methods for IDS based application project:

1. Planning
2. Task development
3. Resource allocation
4. Progress tracking
5. Control

Background: Besides the technical challenges associated with the implementation of an IDS information integration mechanism in an engineering organization, the management of such a project presents several unique challenges. Unlike traditional stand-alone application developments, an IDS implementation must extend over organizational boundaries in order to be effective. Because of the issue of legacy systems, an IDS implementation will also extend over many years. Thus, the management of such a project must take a long range view relative to technology decisions, cost / benefit projections, and design strategy decisions. In fact, an IDS implementation must be treated as the establishment of an ongoing program, requiring the support of the highest levels of management in the organization. There exists major education responsibilities and considerable professional risk for the IDS implementation management. This is due to the fact that (as pointed out by John A. Zachman) "One can readily delineate the merits of the large, complex, enterprise-oriented approaches. Such systems allow flexibility in managing business changes and coherency in the management of business resources. However, there also is merit in the more traditional, smaller, sub-optimal systems design approach as well. Such systems are relatively economical, quickly implemented, and easier to design and manage." Project management methods within such an environment must take this program context into account.

Approach: The approach for this project is to assemble the methods currently used in stand alone application development project management and (with the assistance of the IDS development and prototype implementation teams) map these existing methods onto the IDS SDP0. From this mapping, a set of voids and deficiencies in the existing methods will be identified. This set of method voids, plus the experience based needs from the IDS implementors, will then serve as the requirements for defining a modified set of methods.

Resources: This project will require 6 man years of effort.

Timing: The estimated time frame for this effort is 16 months.

Constraints: This project can be scheduled to start shortly after the initial tactical planning phases of the IDS integration framework for DoD prime contractors is complete.

3.5.3 System Development Environment Thrust

The projects in this thrust area represent the projects that will result in the software / system factory integration environment (referred to as the SDE) of the IISEE. The SDE provides for the IISEE an information and process integration mechanism similar to what the IDS provides for the business environment. The SDE is the shell with the necessary common utilities to allow "framework" directed use and integration of the component tools as well as the knowledge based tools in support of an IDS implementation and evolution. Figure 3.5 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation.

The following sections provide a description of each of the thrust area projects.

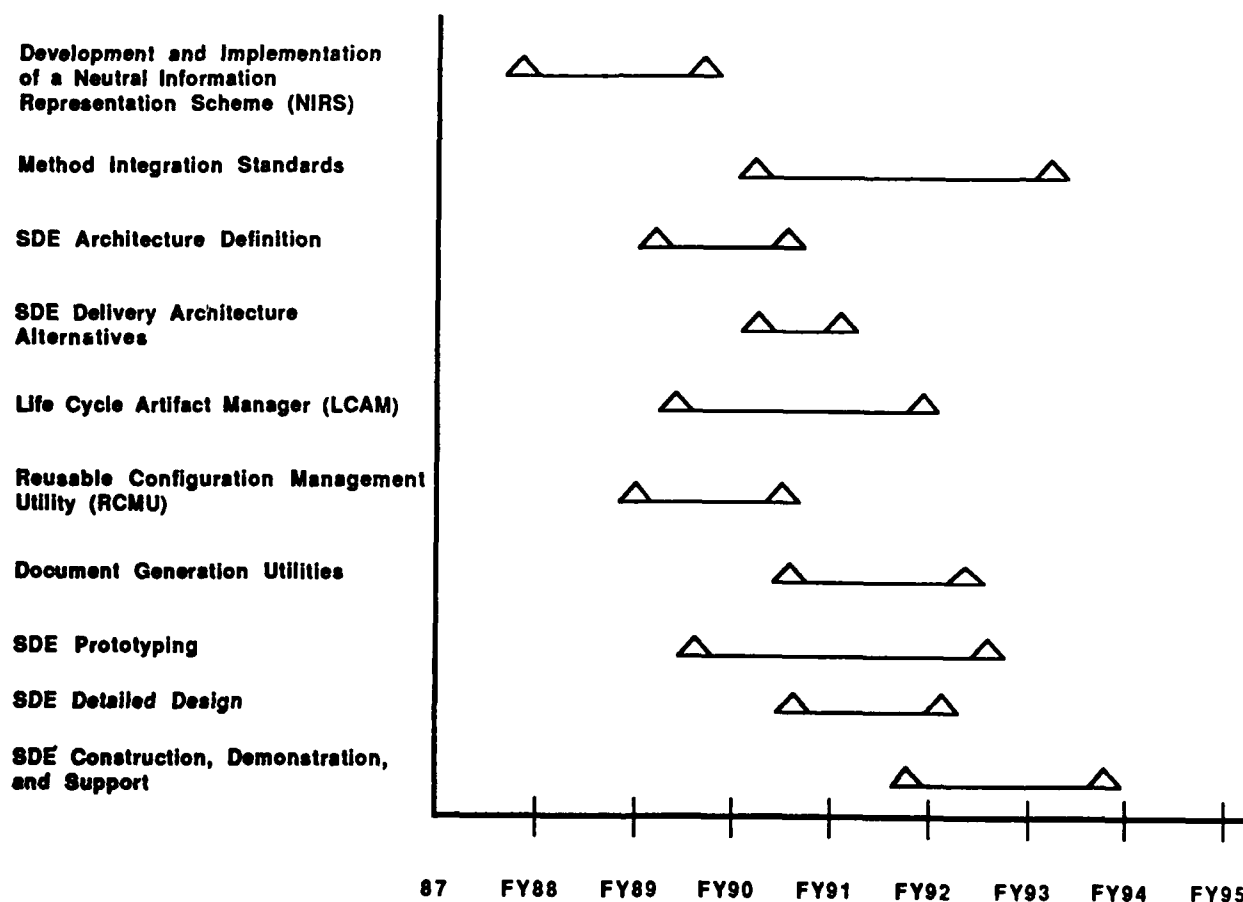


Figure 3.5: Computerized Environment Development Thrust

Project Title: Development and Implementation of a Neutral Information Representation Scheme

Project Goals: The goal of this project is to develop a flexible representation mechanism to support the integration of various component modeling methodologies employed within an IISEE framework.

Project Objectives: The objective of this project is to apply the formalization results generated in the method formalization project to the definition, design, and implementation of a neutral information representation scheme (NIRS).

Background: The NIRS is intended to be a computer processable medium for representing the results of application of the component methods associated with an IISEE framework. It might better be thought of as a knowledge representation language that is so designed as to be able to represent the knowledge about both the "AS IS" system and the evolving "TO BE" system. Its primary purpose is to support the movement of information:

1. Between methods in a particular methodology (e.g. from ENALIM to IDEF1 or IDEF1-X).
2. Between methods in different methodologies (e.g. from IDEF0 to IDEF1-X).
3. Between methods used in different phases of the life cycle (e.g. from an IDEF1-X to a Data Structure Diagram).

So envisioned, the NIRS will provide a modeler with the capacity to store information gathered by any major modeling technique in such a manner that it can be used in other stages of the system life-cycle and it can be displayed in the form of alternative modeling techniques.

Approach: The approach suggested for this representation scheme development is a two phased approach. The task of the first phase is to analyze the formalization of the information and function methodologies and design data structures for capturing the common semantics between the methods in each methodology. After a core set of these representation schemes has been established for the common semantic components, individual specific schemes would need to be put in place and translation heuristics developed to map between the views of the common information. It is recommended that initially the heuristics for mapping between IDEF1 and ENALIM or IDEF1/X be attempted and as the mechanisms for implementing such heuristic strategies evolves, shift the focus to mapping between IDEF0 and IDEF1 or IDEF1 and the process and data state modeling methods.

The second phase of this project would evaluate the prototypes developed in the first phase and attempt to methodize the extension of the NIRS to allow for a structured approach to its future enhancement as new methods are introduced and as new mappings are required.

Resources: The estimated costs for phase I of this project is 5 man years. The estimated cost of phase II is 8 man years.

Timing: The timing for phase I of this effort is 12 months, and 9 months for phase II.

Constraints: This project must be coordinated with the method formalization project and the life cycle artifact manager project.

Project Title: Method Integration Standards

Project Goals: The goal of this project is to establish the working mechanisms and foundations for industry standards to ensure future integration of new methods and tools.

Project Objectives: The objective of this project is to form an industry working standards group to define a standard for information exchange between manual methods and the automated tools which support those methods.

Background: One of the critical needs identified by the industry survey and the IDS development team is the lack of standardization in terminology and semantics across the set of existing methods and tools. The method formalization project and the NIRS project will isolate the base concepts in the existing methods. This project would be tasked to establish usable guidelines for method and tool developers to extend the base concepts or re-engineer them in consistent manners. This project would also be tasked with the establishment of a data exchange standard for moving information between automated tools.

Approach: This project has both a strong technical and strong political aspect to its implementation. It must be guided by the technical results of the method formalization effort and at the same time sensitive to the commercial tool base. Since many of the existing tools are marketed as unique and complete solutions, the only reasonable way to establish data exchange standards between either the manual or automated products is for such a standard to be demanded by the users of these tools. Thus, the first phase of this effort must be focused on organization and education of the users and those commercial vendors who can be convinced of the long term benefits to the industrial and government community. Once this organization is in place, the results of the formalization and NIRS development can be examined as a basis for the development of the necessary standards.

Resources: The estimated cost for this project is 6 man years.

Timing: The time frame for this effort is 36 months.

Constraints: This project obviously must follow the lead of the method formalization project and the NIRS development project. In order to remain consistent with the concept of the standards being user driven, this project should also follow or at least parallel the integration framework development tasks as these will develop a base of IDS users.

Project Title: System Development Environment (SDE) Architecture Definition

Project Goals: The goal of this project is to establish the design of an integrating environ-

ment to support the delivery of a complete life cycle complement of automated framework and method support tools.

Project Objectives: The objectives of this project include:

1. Definition of the requirements for an SDE.
2. Development of a conceptual design of the SDE.
3. Demonstration of a prototype SDE.

Background: As illustrated in Section 2.4 of this report, the SDE is to the IISSE as the IDS is to an organization. That is, it represents the kernel environment for management and control of the information produced as life cycle artifacts throughout the IDS implementation and evolution. Therefore, it must be tightly integrated with the IDS structure. In fact, a working prototype of this kind of mechanism (the ICAM ISDS) was structured as a three schema architecture implementation.

Resources: The estimated resource requirement for this project is 5 man years.

Timing: This project can be completed within 16 months.

Constraints: Must follow the IISSE architecture definition.

Project Title: SDE Delivery Architecture Alternatives

Project Goals: The goal of this project is to define a set of hardware configurations and minimum hardware requirements for the practical delivery of an SDE support environment for the various IISSE implementation contexts.

Project Objectives: The objective of this project is to identify the minimum hardware and support software required to field the SDE capabilities in the various organization types for which frameworks are being established. A second objective of this project is to develop a method for determining of an organization's hardware requirements for SDE and an approach for evolving from one level of SDE implementation to a higher or lower level.

Background: The support requirements which the hardware and software components of an SDE must address will obviously vary based on the size of the organization and the information integration strategy which that organization is pursuing. Therefore, alternate hardware and systems architectures must be defined to cost effectively support the various levels of needs. These architectures must be compatible with the IDS hardware and software support architectures so that the SDE can effectively serve as the vehicle for feeding those systems with new conceptual / internal / external schemas, mappings, and interfaced / integrated applications. A methodology must be established for use by an organization to estimate the level of support required so that the appropriate capitalization can take place. However, since the recommendations of that methodology will not likely be followed

because management will attempt to minimize their risk exposure, an evolution method must be established so that the initial architecture can be expanded or replaced with minimal disruption to the IDS implementation / evolution.

Resources: The resources required for this project is estimated at 3 man years.

Timing: This project would require 10 months to complete.

Constraints: This project should follow the initial SDE architecture definition.

Project Title: Life Cycle Artifact Manager (LCAM)

Project Goals: The goal of this project is to develop a knowledge base management system capable of supporting the intelligent storage, retrieval, evolution, and control of the information generated during an IDS implementation program from the initial strategic plans to the operational software systems.

Project Objectives: The objective of this project is to implement a knowledge base manager based on the NIRS which can address the SDE information storage retrieval and control requirements to support the integration frameworks defined for IDS implementation in the industrial and logistics center environments.

Background: Besides the storage of common model data the SDE must manage the complete complement of life cycle information including needs, business strategic plans, critical success factors, information strategic plans, business rules, data control rules, etc. Some of this information will be stored in separate data or knowledge bases and the SDE will only have to provide access and control of that information. In other cases, there will be a need to provide sophisticated knowledge representation support for information which will be accessed by a wide range of method support tools and decisions makers. This project is responsible for the development of a knowledge based management system which meets the SDE requirements. It is anticipated that the emerging object oriented data base managers may be robust enough to fulfill these needs. It is also recognized that the IDS ECS and DAS may in the future need to provide a similar level of functionality, this offers the possibility of joint development of this utility.

Approach: The approach for this development follows the traditional spiral life cycle development process with a planned three iterations through the development cycle. The first phase will examine the direct use and / or extension of existing capabilities such as the Ontologics VBASE product, the Carnegie Group Knowledge Craft product, or the MCC Proteus product. The second phase would focus on the evolution of an existing capability or the initiation of a full scale development of a new system based on the findings from phase I. The last phase would focus on the implementation issues and performance tuning problems on the SDE hardware suite identified in the previous project.

Resources: If carry over of an existing commercial product can be achieved, the estimated cost for all three phases of this project is 10 man years. If a new concept must be implemented

from scratch, this estimate should be tripled.

Timing: The time frame for phase I is estimated at 10 months. The time frame for phase II is 8 months assuming only modifications of an existing knowledge manager; otherwise, it will require 15 months. Phase III should require (in either case) 12 months.

Constraints: This project should logically follow the SDE architecture design and the initial framework development projects.

Project Title: Reusable Configuration Management Utility (RCMU)

Project Goals: The goal of this project is to provide the a reusable configuration management utility which can be applied across all of the life cycle artifacts.

Background: A key component in the SDE is a consistent, reusable configuration management and control utility which can be applied to the complete spectrum of life cycle information objects as dictated by the integration framework for a particular IDS implementation effort. This implies the capability to be programmable with the particular management rules specific to the object type, life cycle phase, and organizational situation. This utility must be integrated with the LCAM, the IDS ECM, EDI, and GDM. It is anticipated that the emerging object oriented data base managers may be best suited as a basis for the development of these needed capabilities.

Resources: This project will require approximately 10 manyears of development resources.

Timing: The development of this component should be achievable in 18 months.

Constraints: The timing of this project is set to start after the completion of the IISEE design project.

Project Title: Document Generation Utilities

Project Goals: The goal of this project is to take advantage of the completeness of the system description information contained in the SDE and associated component tools to reduce the labor intensive production of textual documentation.

Project Objectives: The goal of this project is to develop a document generation system based on natural language text generation techniques to support the automatic text generation from the contents of the LCAM.

Background: While the installation of an SDE and the component methods associated with the IISEE would largely eliminate the need for paper documentation in the IDS implementation environment, there will always exist the need for production of the paper version. However, the availability of text generation techniques will allow a large portion of the manual effort required for the production of this data to be eliminated.

Approach: The approach recommended for this project consists of a two phase effort. The first phase will be focused on identification of the types of documentation typically required and the modeling of the information content of each type. During this first phase the experience and techniques used in the AFHRL Tech Order Authoring system should be evaluated and that products capabilities reviewed as a base for extension to cover these needs. The second phase will focus on the design of the mapping of the SDE information to an intermediate form suitable for driving a document generator. The third phase will prototype a document definition system and a document generator driven by the heuristics generated for a particular document type.

Resources: The resources required for phase I of this project are estimated at 3 man years. The resources required for phase II of this project are estimated at 2 man years. The resources required for phase III of this project are estimated at 4 man years

Timing: The total time for all phases of this project is 24 months.

Constraints: This project must at least follow the LCAM prototyping phase.

Project Title: SDE Prototyping

Project Goals: The goal of this project is to introduce the SDE concept as early as possible into the IDS development and implementation projects for evaluation and experimental use.

Project Objectives: The objective of this project is to develop a series of prototype SDE implementations which would be provided to the IDS development team and the integration framework efforts for evaluation and testing to evolve the functionality and user interface requirements for the IDS.

Background: Consistent with the information engineering approach followed in the IDS development, it is necessary to prototype the SDE to enable use and feedback from the system development community. It is recommended that the first prototypes be constructed on top of the integrated programming environment of a Lisp machine. In the latter stages the kernel functions will have to be ported to more traditional architectures unless the Lisp chip technology emerges as co-processors on traditional application delivery machines.

Approach: The approach to this project is to produce a prototype SDE every year for three years. The first prototypes will serve as development platforms for the SDE components in a boot strapping fashion. At least one version of each of the component tools will be developed in these prototype environments. These prototypes will also be made available to the IDS developers and IDS implementation efforts under the integration framework development activities.

Resources: Each SDE prototype will require at least 8 man years of development effort. An additional 3 man years per year is planned in the last two years for user support of the previously distributed systems.

Timing: This project is expected to last for 36 months.

Constraints: The development of the first prototype can be scheduled to proceed immediately following the requirements definition phase of the SDE architecture project.

Project Title: SDE Detailed Design

Project Goals: The goal of this project is to establish the specifications of a production SDE.

Project Objectives: The objective of this project is to develop the detailed design for the SDE based on the results of the requirements and prototyping efforts.

Background: This project includes an implicit recommendation that the specification of the production version of the SDE be developed in conjunction with the SDE prototyping efforts but as a separate project to allow the design team to make maximum use of the prototype experience while maintaining the necessary independence to make structural deviations required for a delivery system.

Approach: This project is scheduled to start in the middle of the second SDE prototyping effort. At this point, it will be able to borrow concepts from not only the SDE prototype but at least two versions of the IDS and the ICAM IISS implementation. It would employ the detailed design methods established in the generic IISEE framework definition.

Resources: The planned resources for this project are 3 man years of effort.

Timing: This project is expected to take 12 to 15 months.

Project Title: SDE Construction, Demonstration and Support

Project Goals: The goal of this project is to complete the development of a production version of the SDE and to demonstrate the effectiveness of that product.

Project Objectives: The first objective of this project is to realize a production version of the SDE and to implement that capability in an IDS implementation environment. The second objective of this project is to support the integration of the evolving component tools. The third objective is to demonstrate the effectiveness of the SDE to support an IISEE integration framework in an IDS implementation program. Finally, this project is to provide continued support for the users of SDE over the remaining life of the IISEE program.

Approach: A suggested approach to this project would involve three phases of effort. The first phase would be responsible for the construction, integration and testing of the SDE kernel and the core components under development in this thrust. The second phase would support the integration of the individual method support tools and the establishment and support of a user test community. The third phase would combine maintenance distribution and continued enhancement of the proven configuration.

Resources: The realization phase of this project is estimated at 20 man years. The initial implementation and modification effort is estimated at 15 man years. The maintenance / enhancement and support of the SDE for the remaining years of the IISEE will require at least 5 man years per year.

Timing: The construction of the SDE will require 16 months, the initial component tool integration and test site implementation will overlap the construction effort by four months and require an additional 10 months. The support activities are planned for the remainder of the life of the IISEE.

Constraints: Project starts after detailed design of SDE.

3.5.4 Component Tools for Information Evolution Management Thrust

The projects in this thrust area represent the critical component tool developments necessary to support managers in realistic IDS implementation environments. This thrust also includes the standards development necessary to make use of commercially available tools in the IISEE software / system factory in support of project management activities. Figure 3.6 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation.

The following sections provide a description of each of the thrust area projects.

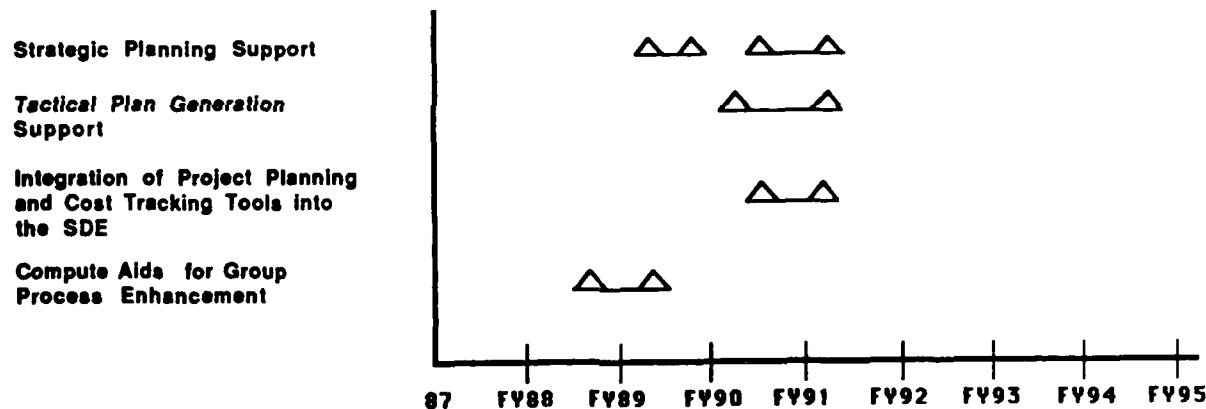


Figure 3.6: Component Tools for Project Managers Thrust

Project Title: Strategic Planning Support

Project Goals: The goal of this project is to provide managers with the necessary automated support to effectively carry out an IDS strategic planning task.

Project Objectives: The goal of this project is to establish an integrated set of information collection, analysis, and aggregation tools for support of the strategic planning process. The integration must allow continued management of the strategic plans and assessment of change impact over time.

Background: Strategic planning tools are required to allow the accumulation of individual organizational goals and the summarization of these goals at the overall corporate level. A capability is needed to manage the mapping of these goals to a set of critical success factors for each organizational unit. An indexing capability is needed to allow identification of common success factors and goals across organizations and a means of mapping information system strategies to these common directions. Since the integration program will extend over a long period of time, there must be support for the determination of the change impact of goals and success factors on the information system strategy and the information system design decisions. This implies the need for these strategic planning tools to be integrated with the LCAM and for the system design support tools to allow for the definition of "influences", "causes", and "dependency" relations between the information strategies and the system design structure and composition.

Approach: The recommended approach to this tool development is to use a rapid prototyping method building off of a robust knowledge base management system such as the Knowledge Craft CRL product from Carnegie Group or the NoteCards system from XEROX. These tools support the dynamic creation of complex relation types, easy prototyping of sophisticated user interfaces, and the ability to formulate complex queries against the information in the knowledge bases. The development process could then proceed in much the same fashion as a knowledge engineering process with the rapid development of prototype tools and the evaluation of these tools by experienced strategic planners.

Resources: This project is estimated to require six manyears of development effort.

Timing: The time period for the prototyping effort of this project is estimated at three to six months. The refinement of the prototype into a fieldable system will require an additional twelve months.

Constraints: This project should follow the intial phases of the prime contractor integration framework development activities.

Project Title: Tactical Plan Generation Support

Project Goals: The goal of this project is to provide support to the project manager in an IDS implementation program for dealing with the complexity of the tactical planning problem.

Project Objectives: The objective of this project is to develop a tactical planning capability to assist managers in the development of the project plans necessary for an IDS implementation program.

Background: Tactical planning is essentially a complex design process. It requires the design of a set of activities and a time allocation of resources to accomplish the strategic goals. Existing PERT and CPM techniques are useful analysis tools but lack the design support capabilities to cost effectively generate a complex IDS implementation plan. What is needed is a set of tools which are integrated with the strategic planning, requirements analysis, and architecture design tools. This set of tools must support the partitioning of the implementation architecture development into affordable development projects. The tools must also support the project planning for the transition of the legacy systems into the integrated environment. The tools must have the capabilities to be integrated with traditional program planning tools so that the results of the tactical planning can be easily transitioned into the individual project management environments and so that changes to the tactical plan can be evaluated for the impact on ongoing projects.

Resources: The resources required for this project are estimated at eight man years.

Timing: The time period for this project is estimated at eighteen months.

Constraints: This project should not be started until the prototyping portion of the strategic planning project is complete.

Project Title: Integration of Project Planning and Cost Tracking Tools into the SDE

Project Goals: The goal of this project is to provide the basic working managers tool set integrated with SDE.

Project Objectives: The objective of this project is to develop a traditional set of project planning, budgeting, scheduling, and cost tracking tools as an integrated set of components within the SDE.

Approach: The recommended approach to this project is summarized in the following tasks:

1. Identify existing project planning, budgeting, scheduling, and cost tracking automated tools.
2. Define the requirements for integration of those tools within the IISSE environment.
3. Develop the necessary software for the integration of these tools into the SDE.

Resources: The resources required for this project are estimated at 4 man years.

Timing: The analysis and development time required for this project is 12 months.

Constraints: This project should not start until the IDS applications project management methods are developed.

Project Title: Computer Aids for Group Process Enhancement

Project Goals: The goal of this project is to develop the methods and multimedia tools necessary to enhance the efficiency of component modeling methods which require the interaction of large groups of domain personnel.

Project Objectives:

Background: All of the existing methods for strategic, tactical and operational planning, analysis, and design require the acquisition of data from a large number of individuals, group review, and group consensus of the method results. Development of tools to structure that group process or change the way the individuals can be brought together to reach consensus descisions will radically reduce the costs of performing the required analysis and modeling.

Resources: The estimated resource requirement for this project is 10 man years.

Timing: The time period required for the development and evaluation of these methods is 12 months.

Constraints: None.

3.5.5 Component Tools for Analysis and Engineering Thrust

The projects in this thrust area represent the critical component tools development needed to support the analysis and engineering activities necessary for an IDS implementation and evolution. Figure 3.7 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation.

The following sections provide a description of each of the thrust area projects.

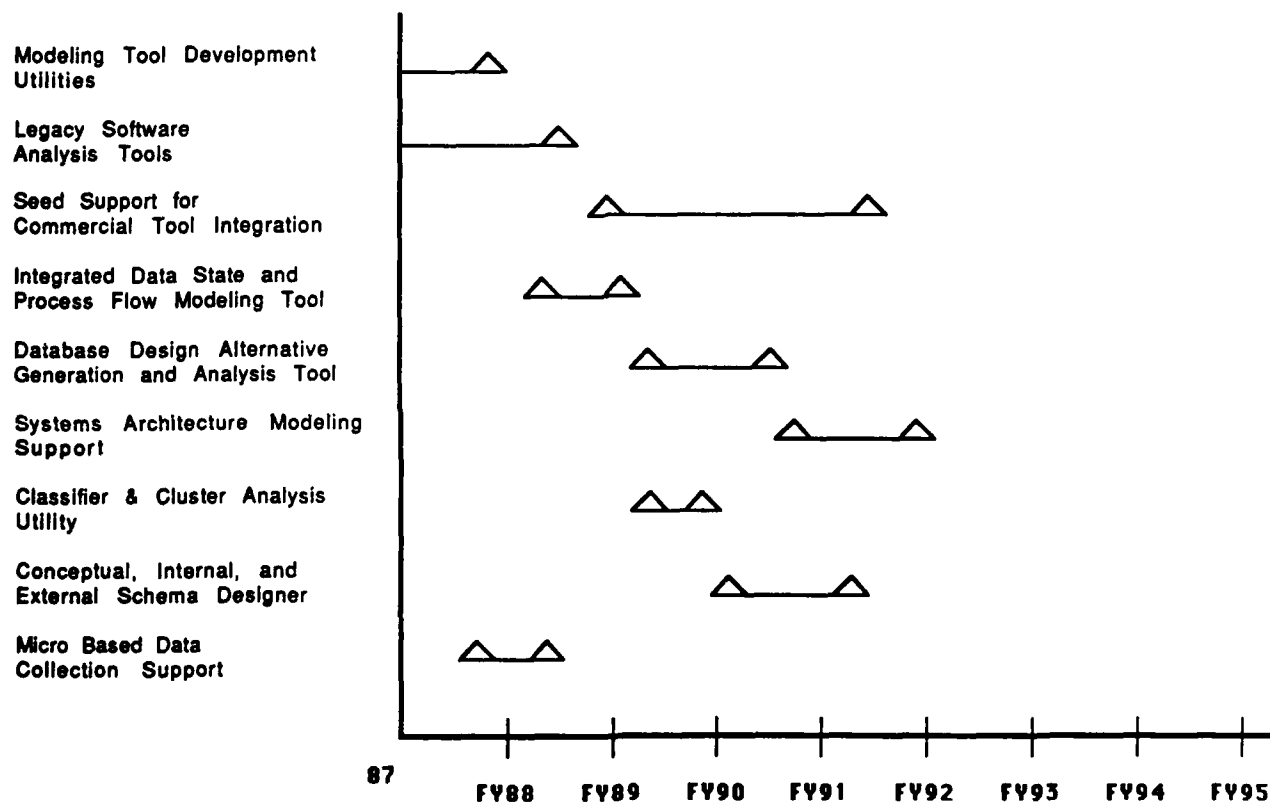


Figure 3.7: Component Tools for Analysis and Engineering Thrust

Project Title: Modeling Tool Development Utilities

Project Goals: The goal of this project is to reduce the time and cost necessary to build modeling support tools for a manual method.

Project Objectives: The objective of this project is to develop a set of modeling support tool generation utilities to allow the development of interactive computer based modeling support utilities (such as an AutoIDEF0) with less than six manweeks of effort.

Background: The large number of different methods with alternative display presentation forms must be employed to successfully carry out an IDS implementation program warrants the development of a flexible utility for the development of computer aided modeling support tools.

Resources: The resources required for this tool generation capability development are estimated at 6 man years.

Timing: The time required to carry out this development is estimated at 12 months.

Constraints: There are no specific constraints on the initiation of this task.

Project Title: Legacy Software Analysis Tools

Project Goals: The goal of this project is to provide a comprehensive set of software analysis tools for assisting in the evaluation of the exiting software base during an IDS implementation.

Project Objectives: The objectives of this project include:

1. Development of software audit tools to extract the data constraints enforced by a set of modules in a database application system.
2. Modification of existing "call tree" generators for population of process models of a system.
3. Development of data definition trace utilities to map out compiled and included data structures.
4. Development of data base schema processors for generation of partial conceptual schema structures from the existing internal schemas.

Background: By far the most expensive part of the "AS-IS" analysis required to implement an IDS is the definition of the conceptual view of the legacy information systems. Often the original developers of these systems are no longer available or have long forgotten the actual entities managed and the business rules enforced by these systems. Most of the effort at the definition of these systems has focused on the modeling of the external user view of the system and then the patching of a demonstratable interface via complex mapping rules.

Approach: Since these analysis tools will be highly environment dependent, it is recommended that an initial development effort focus on the development of specific tools for a specific application (e.g. IDS implementation at Rockwell). The second phase of the program could use these initially hard coded tools as a basis for generalization to support the development of a configurable set of utilities for building specific analyzers.

Resources: The resources required to develop these analysis tools is estimated at 15 man years. The resources required to build the generalized utilities is 20 man years.

Timing: The overall time period for this project is estimated at 28 months.

Constraints: This project could start immediately.

Project Title: Seed Support for Commercial Tool Integration

Project Goals: The goal of this project is to accelerate the integration of commercial tools into the SDE framework.

Project Objectives: The objective of this project is to provide seed monies for the modification of existing commercial tools to conform to the tool data exchange standards and the SDE framework.

Background: Due to the currently limited market for information integration engineering tools, many of the existing commercial tools are supplied by small business concerns who do not necessarily have the ready resources or the developer base to make the modifications necessary to commit to a new standard. In order to accelerate the acceptance of the tool data exchange standards, it is recommended that seed monies and assistance be provided for the updating of the product documentation with a formalization section and the modification of the tools to provide access to a model in the standard format.

Resources: The resources required for this project would vary depending upon the complexity of the tool.

Timing: This project is planned for a three year time frame.

Constraints: This project could start towards the end of the formalization effort and roughly one year into the standardization effort.

Project Title: Integrated Data State and Process Flow Modeling Tool

Project Goals: The goal of this project is to develop an automated support tool for the data state and process flow methods developed under the Component Method thrust.

Project Objectives: The objective of this project is to produce near term automated support for the application of the data state and process flow modeling method which are integrated with an IDEF0 and IDEF1 modeling capability.

Approach: The approach recommended for this project is to utilize the Lisp machine processing environment to develop a near term automated model development support environment for IDEF0, IDEF1, Data State, and Process Flow modeling.

Resources: The resources required for this tool development is three man years.

Timing: This project would require 6 months to complete.

Constraints: This project would provide a much needed modeling support capability and should be started immediately.

Project Title: Database Design Alternative Generation and Analysis Tool

Project Goals: The goal of this project is to provide an analytic basis for evaluating the impact on database design of the desired data control policies of an organization.

Project Objectives: The objective of this project is to develop a capability to simulate various database design concepts based on the information model semantics, the data state transitions, the process flow characteristics to predict the performance impact of the transaction rates, and integrity rules implied by these models and a particular database design approach.

Resources: The resources for this tool development is estimated at 7 man years.

Timing: This project will require 16 months to complete.

Constraints: The above referenced component methods development should be complete prior to the initiation of this task.

Project Title: Systems Architecture Modeling Support

Project Goals: The goal of this project is to develop a computer aided information system architecture design support tool.

Project Objectives: This tool must provide support for:

1. Easy configuration of system architectures from standard components.
2. Stochastic as well as simulation analysis.
3. Ability to capture the design rationale as well as the design configuration.
4. Modeling of all system interfaces.
5. Display of "AS IS" and "TO BE" architectures.

This tool must implement the information systems architecture specification method developed under the component method thrust.

Approach: The first phase of this effort will be focused on the development of a modeling support tool for the information system architecture definition method developed under the component method thrust. The second phase of this effort will add to that capability support for design analysis via simulation and stochastic modeling.

Resources: This project is estimated to require 5 man years of development effort.

Timing: The development of the modeling support should require 6 months, the analysis capability will require an additional 12 months.

Constraints: The availability of the formalized version of the information system architecture specification language.

Project Title: Classifier and Cluster Analysis Utility

Project Goals: The goal of this project is to provide a set of utilities which can be applied to various life cycle artifacts to allow similarity measures and family groupings to support planning, analysis, and design decision making.

Project Objectives: The objective of this project is to develop a set of tools for factor analysis, characteristic based cluster generation, and fuzzy association to support planning, analysis, and design decision making.

Background: The ICAM projects in Group technology and Integrated Decision Support Systems developed a number of utilities for cluster analysis, fuzzy associations, and characteristic based family generation. This project would build on that base to provide a similar set of tools to be used in the needs analysis and project planning activities within the IISEE.

Resources: This project would require 3 man years of development effort.

Timing: The time required for development of these utilities is 9 months.

Constraints: This project should be coordinated with the strategic and tactical planning methods developments, and the IDEF use procedure developments.

Project Title: Conceptual, Internal, and External Schema Designer

Project Goals: The goal of this project is to reduce the time and cost associated with definition of the requisite definitions for an IDS implementation.

Project Objectives: The objective of this project is to develop a design support tool for the translation of information models, data state models, and process models into conceptual schemas for the population of the IDS PCDM. This also includes support for the development of the translations (or mapping specifications) from the external schemas to the conceptual

schemas and from the conceptual schemas to the internal schemas. Finally this project is tasked with the development of a design support environment which would support the development of user interface presentations and external schemas from the information models, process flow models, and user models.

Background: Essentially this project is a project in design automation for IDS implementor support. There is a related project in the knowledge based systems thrust which addresses the addition of intelligent support to the design automation task. This project is focused on use of traditional methods at least at the time of this conceptualization. It may turn out that heuristic based techniques are required for the various mappings involved.

Approach: The suggested approach for this task is to set up a two phase project. The first phase would be a prototyping phase with the focus of working with the IDS developers and iteratively evolving acceptable solutions to the design support problem. The second phase of the project would be focused on a wider evaluation of the tools by IDS implementers and then, based on the total set of requirements, the development of a production tool set.

Resources: The resources estimated for this task are 7 man years for phase I and 10 man years for phase II.

Timing: Phase I can be complete in 10 months. Phase II will require an additional 12 months.

Constraints: This project should follow the schema mapping method development project.

Project Title: Micro Based Data Collection Support

Project Goals: The goal of this project is to develop a suite of micro based information acquisition support utilities for support of the various methods required for IDS implementation.

Project Objectives: The objective of this project is to develop micro based support for collection, organization, management, and limited analysis and display of information needed to build IDEF0, IDEF1, Data State, and Process Flow modeling techniques.

Background: A large amount of the effort involved in strategic planning for an information integrated program within an organization is the "field" work associated with information acquisition and manipulation. These proposed tools would provide a capability to utilize portable automation to improve the efficiency and productivity of that process.

Resources: The resources required for this effort are estimated at 3 man years.

Timing: This effort can be completed in 8 man months.

Constraints: The data state and process flow method disciplines must be complete prior to the initiation of this task.

3.5.6 Component Tools for Software Development Thrust

The projects in this thrust area represent the critical component tool developments and standards developments necessary to use commercially available tools in the IISEE software / system factory to support the software development process. Figure 3.8 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation.

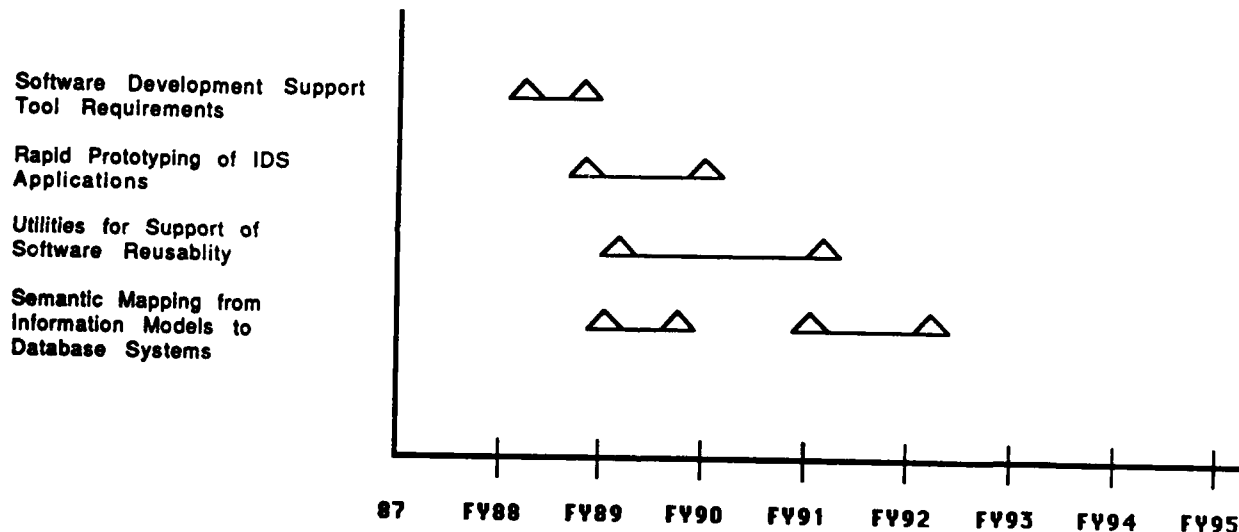


Figure 3.8: Component Tools for Software Development Thrust

Project Title: Software Development Support Tool Requirements

Project Goals: The goal of this project is to identify the productivity leverage areas for automated support to IDS application developers.

Project Objectives: The objective of this project is to define the needs and requirements of the software programmer for specialized programming, integration, and testing tools to support IDS implementation.

Background: The experience base on which this IISSE program plan was structured was heavily influenced by planning and analysis needs. IDS represents a new programming paradigm in the engineering, manufacturing, and logistics computing worlds. The experience which will be accumulated over the next several years will identify a number of low cost tools which offer a high productivity return for the programming staff charged with building and maintaining applications in this environment.

Approach: A traditional needs analysis and requirements definition process and a tactical plan development are the basic tasks required in the approach to this project.

Resources: The resources required for this project are 2 man years.

Timing: This project should be completed in 8 months.

Constraints: This project should be initiated during the initial IDS prototype implementation efforts.

Project Title: Rapid Prototyping of IDS Applications

Project Goals: The goal of this project is to develop the capability to take advantage of the inherent system definition available in the IDS to allow rapid prototyping of new engineering or manufacturing applications.

Project Objectives: The objective of this project would be to develop application generation software for the complete generation of IDS applications from the user, function, information, and process flow model specifications.

Background: Fourth generation development languages and data base prototyping languages have been developed which produce reasonably sophisticated applications from less complete specifications than the IDS conceptual data model. Therefore it is anticipated that with moderate effort, IDS implementations could support extensive prototype generation capability.

Resources: The estimated resources for this development are 10 man years.

Timing: This project should be achievable in 18 months.

Constraints: This development activity could start immediately.

Project Title: Utilities for Support of Software Reusability

Project Goals: The goal of this project is to develop the capability to support reuse of existing IDS schema definitions, schema mappings, user interface software, or complete applications.

Project Objectives: The objective of this project is to develop the analysis tools and dictionary classification mechanisms required to augment the current IDS definitional capabilities in order to support extensive reusability of the application and software base.

Background: Software reusability has been a long sought goal in the software engineering community. However most existing systems suffer from extremely sparse definitional basis outside of the source code itself. The greatest success with software reusability has been achieved in the object oriented programming environment (e.g. Flavors, SmallTalk, and Loops). This reusability was achieved without such dictionary classification mechanisms by providing mechanisms for easily combining the behavior of many small, specialized objects. This particular option can in a certain sense be duplicated in the IDS environment. However, the legacy software problem limits its applicability. The result is that a combination of classification methods and encapsulation methods will have to be explored.

Resources: The estimated resources for this project are 12 man years.

Timing: The time requirement for this project is 28 months.

Constraints: This project should be delayed until the design rationale capture method is firmly underway.

Project Title: Semantic Mapping from Information Models to Database Systems

Project Goals: The goal of this project is to improve the ability of the analyst to interpret large amounts of unfamiliar database system code.

Project Objectives: The objective of this project is to build an analysis tool for the database programmer to allow the interpretation of database schema structures in terms of the original semantic models from which these schemas were designed.

Background: With the availability of design rationale encoding and the information in the conceptual schema, all of the information is available for the construction of a tool to reverse interpret a set of database structures in terms of the original semantic models. This interpretation to a programmer unfamiliar with a large database design can be difficult or impossible to extract since the mapping of the original meaning of a coherent portion of the original code may be dispersed throughout a data base structure. What may have seemed like minor syntactic modifications of entity or attribute labels by the designer can cause significant confusion to the new programmer. In future versions of this tool, knowledge based heuristic techniques could be employed to detect inconsistencies in the use of labels (multiple meanings for the label "part-indenture," for example).

Resources: This project will require 4 man years to develop the initial prototype semantic

mapping tool and an additional 8 man years for a full production utility.

Timing: The first prototype will require 10 months to construct. The production utility would require an additional 18 months.

Constraints: The feasibility of this tool is dependent on the existence of the design rationale capture method and the capability of the LCAM to represent that information.

3.5.7 Knowledge Based Tool Development Thrust

The projects in this thrust area represent the opportunities for the application of knowledge based systems techniques to the IDS system development / application arena. Figure 3.9 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation.

The following sections provide a description of each of the thrust area projects.

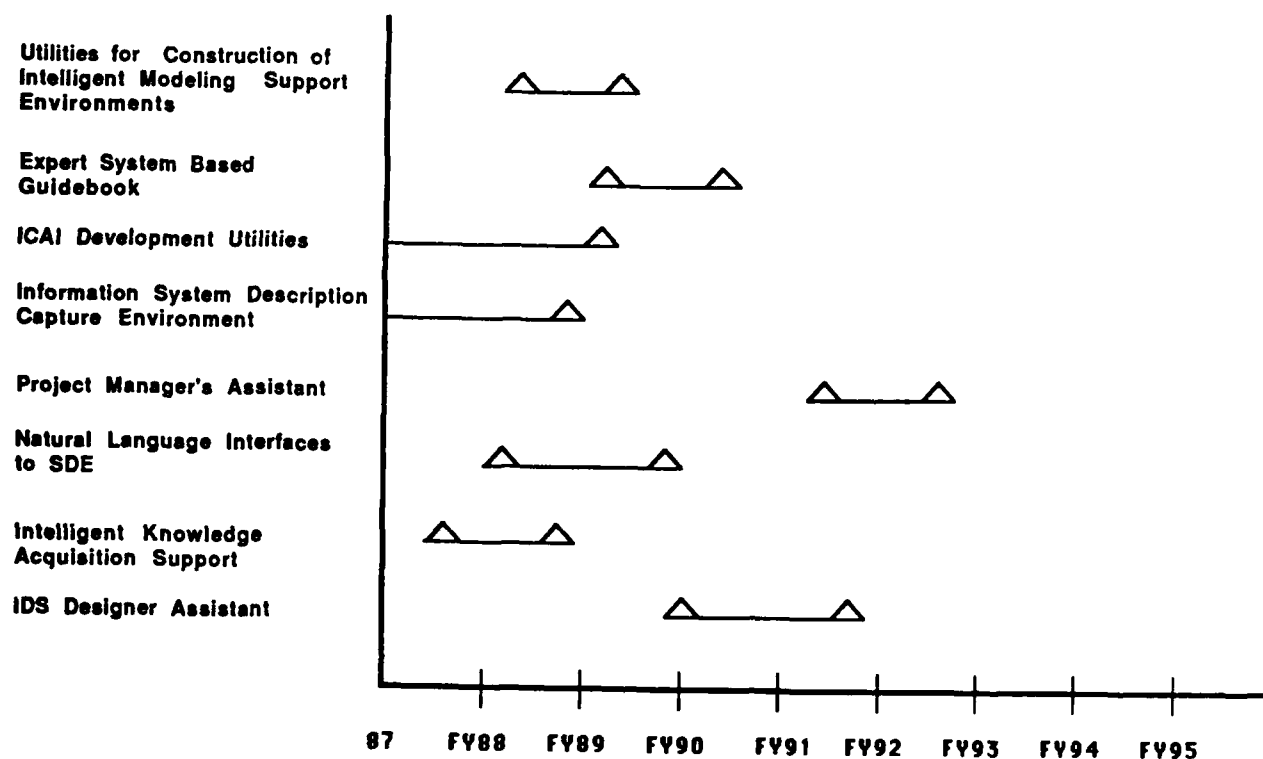


Figure 3.9: Knowledge Based Tool Development Thrust

Project Title: Utilities for Construction of Intelligent Modeling Support Environments

Project Goals: The goal of this project is to put in place the utilities required to build the knowledge based systems described in this thrust area.

Project Objectives: The objectives of this project are to:

1. Define the requirements for intelligent modeling support environment construction utilities.
2. Design the set of utilities to meet these requirements.
3. Construction of the set of utilities.
4. Rehosting of the utilities on the various hardware types in the SDE.

Background: The utilities for constructing modeling support environments provided on modern Lisp machines provide an order of magnitude decrease in the development time and cost for these systems over conventional hardware environments. Most of this productivity increase is due to:

1. The integrated set of editing, compilation, debugging, and user interface construction utilities provided.
2. The object oriented programming environment which maximizes the reusability of the primitives provided.
3. The hardware implementation of the base symbolic manipulation capabilities needed to efficiently support rapid prototyping based on software reuse concepts.

The task ahead of this project is to construct the additional utilities required to achieve the same level of productivity in the process of adding intelligence to the modeling support environments. The research associated with this planning effort has identified the need for the following utilities:

1. Knowledge base management utilities.
2. Reusable reasoning mechanisms including:
 - (a) Constraint propagation mechanisms.
 - (b) Efficient rule production engines.
 - (c) Resolution refutation mechanisms.
3. Libraries of frequently used life cycle artifact objects and relation representations.

Approach: The approach to this project will follow the "Spiral" model of software development. This model calls for the production of an increasingly complex system by iterative application of the traditional life cycle. The planned approach of this project calls for two iterations through that spiral.

Resources: The resources required for this project are estimated at ten manyears.

Timing: The approximate duration of this task is 15 months.

Constraints: There are no project dependency constraints on this utility development project.

Project Title: Expert System Based Guidebook

Project Goals: The goal of this project is to develop the capability to quickly and efficiently orient a new user to the appropriate framework for his environment and to make minor tuning to that framework.

Project Objectives: The objective of this project is to produce an expert system to assist the potential user of an IISEE in the selection of an appropriate framework and the configuration of that framework for his particular needs.

Background: The prior classes of organizational contexts which are of first interest in the development of integration frameworks have been provided in the framework development thrust. However, even within one of these major classes, there is a considerable amount of tuning which could be beneficially applied to a framework to account for the individual site situation. Initially, this tuning will have to be done by hand. The IISEE guidebook will contain the decision procedures for getting a user to the correct framework and the individual framework guidebook will contain the procedure for tuning that framework for the individual needs. It is anticipated that even with this guidance problems due to lack of understanding of the guidebook decision logic will arise. This project proposes to develop a knowledge based implementation of those guidebooks so that the knowledge of the original developers of the procedures can be more directly brought to bear on the situation.

Approach: The approach recommended for this project consists of a two phase effort. The first phase would be focused on the required knowledge engineering and design work required for the development of a proof of concept prototype. The second phase would be focused on the completion of the knowledge engineering and the traditional software development required to produce a fieldable expert system which could be distributed along with the IISEE product.

Resources: This project is estimated to require 7 man years of effort.

Timing: This project can be completed within 16 months.

Constraints: This project is constrained to follow at least the establishment of two of the integration frameworks described in the Integrated Framework Thrust area.

Project Title: ICAI Development Utilities

Project Goals: The goal of this project is to utilize knowledge based systems techniques to assist in the transfer of the knowledge of method experts to the novice participants on a project.

Project Objectives: The objective of this project is to develop a set of utilities required to support the development of intelligent tutoring systems to increase the rate of technology transfer of the IISEE product into production use.

Background: One of the largest impediments to the widespread implementation and use of IDS is the amount of training required of the organizations desiring to employ the technology. Intelligent computer aided instruction or tutoring systems can help to overcome this obstacle. However, ICAI systems have been more difficult and expensive to build than expert systems themselves. Establishment of model driven ICAI development utilities which allow the assisted construction of ICAI applications from student models, domain models, and performance measurement structures could reduce the time and effort for construction of these systems by half.

Approach: The approach to the development of these utilities will follow the "Spiral" model of software development. This model calls for the production of an ever increasingly complex system by iterative application of the traditional life cycle. The planned approach of this project calls for three iterations through that spiral. The first iteration through this spiral would be coordinated with one of the tutorial developments for a component method. This phase would build an ICAI application for this tutorial. The second phase of the project would use the experience gained in the first iteration combined with a requirements study of the other tutorial projects to identify and construct a set of generic utilities. The third iteration would again construct an ICAI application with these utilities and enhance the utilities as required.

Resources: The resources estimated for this project are 15 man years.

Timing: Phase I of this project should require eight months. Phase II of this project will require 14 months and Phase III will require four months.

Constraints: The development of these baseline utilities do not directly depend on previous project results. However, this project should be closely coordinated with the Intelligent Modeling Support Utilities project described above.

Project Title: Information System Description Capture Environment

Project Goals: The goal of this project is to tightly couple the system analysis with the actual engineering environment in order to improve the quality of IDS applications.

Project Objectives: The objective of this project is to establish the capability to automatically generate system analysis models from descriptions of the engineering or manufacturing domain.

Background: The nature of system analysis models is that they are filtered through the interpretation of the modeler. While the users are typically called upon to review and validate the models, they often overlook the implications of the structures or assertions which are made in those models. Often the sheer volume of information which the analyst must process causes him to gloss over key points in the system description.

Approach:

Resources: This project will require 10 man years of development for the capture portion of the system and an additional 8 man years for the model design generator.

Timing: The total project time frame for this effort is 24 months.

Constraints: This project could start immediately.

Project Title: Project Manager's Assistant

Project Goals: The goal of this project is to apply knowledge based techniques to the creation of a capability to leverage the skills of the IDS implementation and application project managers.

Project Objectives: The objective of this project is to develop a knowledge based set of tools for assisting in the planning, design decision making and control tasks of the program and project managers in an IDS implementation environment.

Background: One of the most complex tasks for the project manager in a large development program like the IDS is the management of the design decisions as they evolve through time. Particularly if he must manage several projects simultaneously. Also keeping track of the assumptions on which project plans were built and logically analyzing changes against this assumption base. This project is aimed at the development of a knowledge based assistant. This system would be capable of recording the plan assumptions and performing logical "what if" analysis against that assumptive base. This system would also provide for the recording of management design decisions and the rationale behind those decisions similar to the designers assistant described later in this thrust area.

Approach: The approach recommended for this project consists of a two phase effort. The first phase would be focused on the required knowledge engineering and design work required for the development of a proof of concept prototype. The second phase would be focused on the completion of the knowledge engineering and the traditional software development required to produce a fieldable project assistant system.

Resources: This project will require 12 man years of development resources.

Timing: A prototype version of this tool could be completed in 8 months. A robust version integrated into the SDE tool suite would require an additional 10 months.

Constraints: Work could be started on this tool as soon as the management method development activities were stabilized.

Project Title: Natural Language Interfaces to SDE

Project Goals: The goal of this project is to provide a development capability to the IDS application developer to incorporate restricted natural language capabilities into IDS interface designs.

Project Objectives: The goal of this project is to develop a reusable set of utilities for constructing simple command, query, and text understanding natural language.

Background: The information delivery capabilities of IDS have the potential to far outstrip what the application developer has the ability to foresee or the capability to support prompt and menu interfaces. Restricted natural language (if the user is properly trained in its use) offers the capability to take full advantage of the inherent IDS information potential. However, without the proper development tools and utilities, the construction of a simple command completion, DWIM (do what I mean) or other restricted natural language interface difficult and expensive. Tools (such as the Carnegie Group Language Craft) have been shown to allow the construction of relatively sophisticated natural language like interfaces in short periods of time. In this project we are suggesting the development of a set of utilities which would be fully integrated into the user interface development package on IDS and would support the development of these types of systems.

Approach: The first task associated with this project is to examine the potential application areas for restricted natural language interfaces in an IDS environment. It would be recommended to construct some "clay model" systems with the available prototyping tools to generate ideas and feedback from both the developer community and the user community. The approach recommended for this development task requires an upfront choice of a linguistic paradigm (lexical functional grammar, generative transformational grammar, case frame etc.) based on the results of the requirements and prototyping activities and the availability of accessible software for the paradigm. Once this decision is made then the design of the structure and components of the utilities can be established and the development of the necessary software performed. One of the design goals of the system would be to incorporate an expert system design support tool as an element to assist system designers who are not familiar with the underlying language theory.

Resources: The resources required for this task are estimated to be 8 man years.

Timing: This project will require 24 months to complete.

Constraints: This project could start immediately.

Project Title: Intelligent Knowledge Acquisition Support

Project Goals: The goals of this project are to improve the quality and reduce the cost of information model development and use.

Project Objectives: The objective of this project is to develop automated knowledge based support for the construction of information models and the analysis of information models

for completeness, consistency, and correctness.

Background: In the exploratory research efforts associated with this project, we discovered that the vocabulary and language structure used to describe business rules in the engineering and manufacturing domains was surprisingly restricted. In three separate information models built by three different coalitions over a period of almost seven years, there were fewer than 70 linguistic verb case structures used. This kind of regularity in the structure and use of natural language in a specific domain is usually indicative of the ability to develop sophisticated user interfaces which can process a large percentage of natural language statements about the domain. In the particular case in point, it is possible to construct a system which would accept natural language statements about the business rules of the organization and produce information models in various forms (e.g. IDEF1, ENALIM, ER, IDEF1/X). It would also be possible to construct textual summarization of the implications (implicit assertions) of an information model to assist in the validation process. Limited capabilities of this latter sort already exist as simple paraphrasing devices. However, these concepts could be greatly expanded to include textual summarizations and interactive question and answer discourse.

Resources: The estimated resources for this project are estimated at 6 man years.

Timing: This project is scheduled for 18 months.

Constraints: This project can start immediately.

Project Title: IDS Designer/Data Administrator Assistant

Project Goals: The goal of this project is to develop a capability to provide design evaluation, documentation, and rationale capture to the IDS application designer and data administrator.

Project Objectives: The objective of this project is to provide knowledge based support to the development of the IDS implementation and design rationale capture method.

Background: A comprehensive design support environment must provide:

1. Control and manipulation of the design process state.
2. Support based on the goal structure of the particular design process instance.
3. Record of the design decisions and the assumptions of those decisions.
4. Rationale for the design decisions.
5. Support for the use of predictive analytic models.
6. Support for the use of qualitative evaluation tests.

With the definitional basis of the LCAM and the IDS conceptual schema, the foundation exists for providing intelligent design support to the IDS implementation administrator and to the IDS application developer in all of the above listed areas.

Resources: The resources required for this project are 12 man years for prototype and validation of that prototype by the IDS design experts.

Timing: This project should take six months for the initial prototype and 15 months of expansion and validation to achieve a usable system.

Constraints: This project should be delayed until the completion of the initial IDS implementations.

3.5.8 Technology Transfer Thrust

The projects in this thrust area represent the technology transfer efforts in support of the IISSE which need to be undertaken. It should be noted that the real objective of this thrust is to transfer understanding. Figure 3.10 identifies each of the projects currently scoped for this area and illustrates the relative timing and sequence of their initiation. The following sections provide a description of each of the thrust area projects.

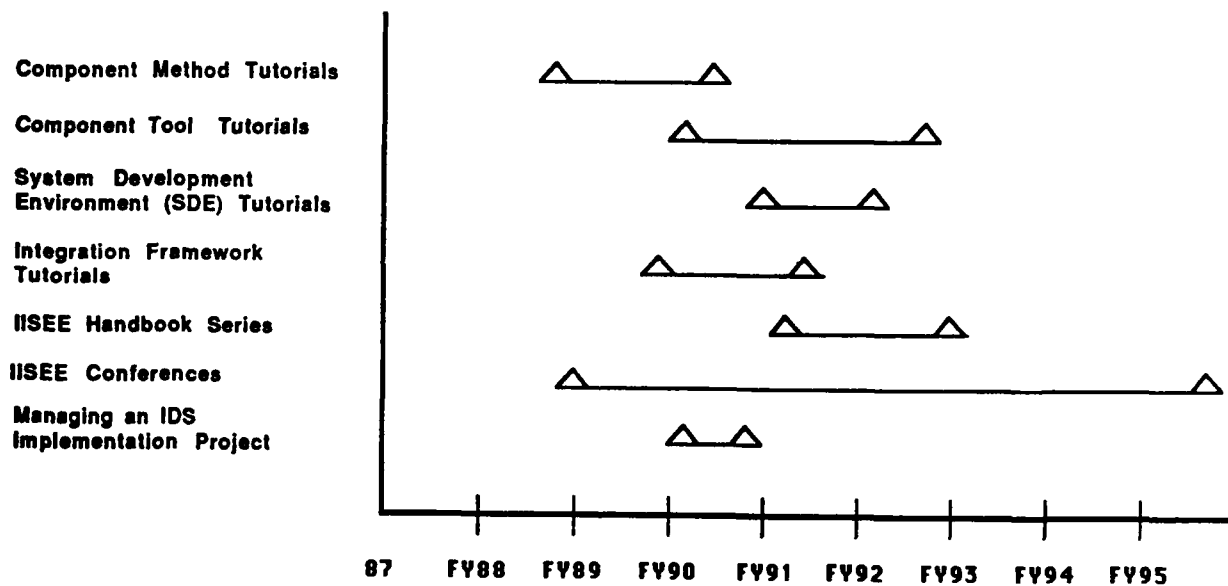


Figure 3.10: Technology Transfer Thrust

Project Title: Component Method Tutorials

Project Goals: The purpose of this project will be to develop a tutorial type of educational package which will provide an overview of the component methods available for systems modeling, procedures definition, systems design logic, system architecture definition, and other pertinent design areas. As a secondary goal, the overview will be supported by individual educational modules which outline in detail the theory, technical qualities, and implementation strategies for each of the individual tools. The long term targets will be to build a large volume of educational modules for technology transfer that will be linked together through an overview or tutorial package that introduces the methods to the student.

Project Objectives: The objective of the proposed project will be to establish a foundation and a structure for a framework overview module and for the modules that detail each individual method. The initial thrust will be to define a flexible structure that can be added to as the frameworks and new methods evolve. The initial deliverables will include a highly structured overview module and three detailed modules illustrating how a total method package will fit together.

Background: To date, training and/or technology transfer packages have been devoted to the education of individuals in the use of a particular tool (i.e., IDEF0, IDEF, group technology). Very little effort has been put into the establishment of a comprehensive package of technology transfer modules that are logically tied together through an overview module. In the area of component methods, it is especially important that system designers be provided with an understanding of available methods, how they can be effectively used and how individual methods relate and/or support each other. For data collection purposes, the knowledge of several methods and how they relate to each other will allow the system designer to gather data with multiple applications in mind. A module which supports the design of data collection instruments should clearly outline for the system designer how to design the instrument to gather data for multiple modeling activities. The rationale behind this project is that the technology transfer efforts for component methods should be centered around the development of a structured set of closely interrelated packages that build upon each other. The individuals involved in technology transfer (training) should have available to them a tutorial or overview that allows them to judge the areas in which they need to be trained to achieve their goals. The tutorial should provide the information required to make informed decisions regarding the modules that should be studied by a user.

Approach: The approach will be to establish a structure or framework that will allow the development of an easily expanded general module that can readily accommodate detailed packages to be connected to it. Once the framework is established, a prototype general module will be written and combined with at least three detailed modules to form a test package. The test package will be brought to the user community and trials run to determine acceptability. The refinement of such a package will be an iterative process and will require a great deal of review and contact with the user community. Each module will be documented in a workbook format and a roadmap like document relating the overview module with the individual modules will be developed.

Resources: This project will require at least one author for the overview module combined with one author for each of the individual modules. Word processing software and some graphics support software will be required.

Timing: A minimum of six months will be required to complete the first phase of this project. Each individual module can be prepared in parallel with the overview model providing the link between them. Such an undertaking will require four months of concentrated writing effort with two months of review and coordination. The second phase will require six months with one person bringing the modules to industry for presentation and evaluation.

Constraints: This project can be started immediately and scheduled to last the extent of the methods development process.

Project Title: Component Tool Tutorials

Project Goals: The goal of this project is to develop the necessary training materials, exercises, and example base for the technology transfer of the component automated tool (both traditional and knowledge based) transition to industrial and government users.

Project Objectives: The objective of the proposed project will be to establish a foundation and a structure for a tools overview module and for the modules that detail each individual tool. The initial thrust will be to define a flexible structure that can be added to as tools evolve and as new interests in tools evolve. The initial deliverables will include a highly structured overview module and three detailed modules illustrating how a total tools package will fit together.

Background: The purpose of this project will be to develop a tutorial type of educational package which will provide an overview of the component tools available for systems modeling, procedures definition, systems design logic, system architecture definition, and other pertinent design areas. As a secondary goal the overview will be supported by individual educational modules which outline in detail the theory, technical qualities, and implementation strategies for each of the individual tools. The long term targets will be to build a large volume of educational modules for technology transfer that will be linked together through an overview or tutorial package that introduces the tools to the student.

Approach: The approach to this project is similar to that for component methods tutorial development with the addition of the investigation into on-line tutorial structures which take advantage of the functionality of the tool itself.

Resources: The resources required for this project are estimated at one man year per tool.

Timing: The time period planned for each tool tutorial development is six months.

Constraints: The needs and design approach for the on-line portion of the tutorial development should be coordinated with the component tool developers.

Project Title: System Development Environment (SDE) Tutorials

Project Goals: The goal of this project is to develop the necessary training materials, exercises, and example base for the technology transfer of the automated System Development Environment (SDE) transition to industry and government users.

Project Objectives: The objective of the proposed project will be to establish a foundation and structure for an SDE overview module and for the modules that detail each individual common utility in the system (i.e. LCAM, RCMU, DCU).

Background: While the individual component methods and tools will have limited classes of users, the SDE (being the automated integration framework) must support interaction by many different classes and levels of users. Therefore specialized tutorials must be developed for each class of user.

Approach: The approach for this project requires the initial development of user types for SDE use. Once these user classes are defined, an approach identical to that used for the component tool tutorials can be applied.

Resources: The resources required for each user type tutorial is one man year.

Timing: The overall development time required for this project is estimated at 16 months.

Constraints: This project should be initiated during the detailed design phases of the SDE development.

Project Title: Integration Framework Tutorials

Project Goals: The IISSE "Frameworks" provide the backbone of the IISSE application in a particular environment. The goal of this project is to develop the training materials associated with each IISSE framework to allow correct use of that framework by management, project administration, and technical personnel.

Project Objectives: The objective of this project is to develop the education and training modules necessary to describe the philosophy, implications, and applications of the IISSE integration frameworks for each role of participant in an IDS application. This objective also covers the education modules necessary to describe the IDS standard definition and how to tailor those concepts to a particular implementation site.

Background: The integration frameworks, like the SDE, must be understood and used by a large class and variety of users. Therefore, specialized tutorials must be developed for each class of user. The guidebook component of the IISSE contains a description of the general philosophy of IDS implementation and application development within that environment. However, each individual participating in the initial IDS implementation or maintenance must specialize those concepts and relate them to his / her activities in the associated framework defined procedure. This adds a level of complexity to the development of training materials for transfer of framework methods. It implies that the training approach will have to be heavily oriented towards case studies and group practice methods.

Approach: The approach for this project requires the initial development of the user roles in each integration framework use. Once these user classes are defined, an approach identical to that used for the component method tutorials can be applied.

Resources: The resource estimate for this task is 3 man years per user type. The additional effort over that estimated for the component method tutorials is due to the case study nature of the framework tutorials.

Timing: This project will require 12 months of development effort.

Constraints: This project should run in parallel with the framework development projects.

Project Title: IISSE Handbook Series

Project Goals: The long term goal of this project will be to establish evolving documentation of the experience base of industry and government organizations in how to achieve information integrated systems.

Project Objectives: The objective of this project is to bring together experts in the field of integrated systems design and develop a series of textbooks. Due to the diversity of such a set of authors, the books would have to be written as a handbook under the guidance of an editor.

Background: Many experts in the field of integrated systems design have written papers and reports of significant note. These papers appear in a far reaching variety of journals, conference proceedings, and company reports. There is a need for a single textbook or handbook in this field. The rationale behind the project is that the technology of integrated design systems can be transmitted to a very far reaching audience of systems people through a textbook.

Approach: The editor will write an outline for the handbook. A selected advisory board of industry and university experts will critique and shape the outline. Once the outline is stabilized with regard to content, authors will be selected. The advisory board will suggest potential authors. Through the board's contribution, the editor's knowledge of the field, and a literature search, authors (contributors) will be named. According to the topic and content of the sections to be written by an individual author, the length of each selection should be between 35-70 pages. All sections of manuscript will be reviewed by industry experts and receive approval before acceptance for publications.

Resources: This project will require an editor and approximately ten authors for the individual sections of the handbook. The time to write, review, and edit such a handbook will be two years. Of most importance in such a project is an administrative coordinator to track manuscripts and to organize the manuscript materials. Assistants (graduate students) will also be of key importance to this type of project.

Timing: As mentioned above, such a project will require a minimum of two years of work for the editor on a one-half time basis. The individual authors will require a wide range of

times depending upon their experience and existing manuscripts.

Project Title: IISEE Conferences

Project Goals: The goal of this project is to establish a forum for industry, academic, and government exchange of ideas and experience in the development of large scale integrated information systems. This project should establish the basis for future developments and extensions to the IISEE to be supported by a professional network of practicing system developers.

Project Objectives: The objective of this project is to organize and operate a yearly technical conference in the implementation of integrated information engineering.

Background: Open technical conferences have long been shown to be one of the most effective mechanisms for technology transfer. This is particularly true of technology which is largely practice and experience based. One of the shortfalls of the previous IDEF and ICAM SEM method development initiatives was that they did not provide this forum for idea exchange. Such a conference would not only provide a mechanism for the exchange of ideas it would also provide a forum for feedback by industry to the IISEE and IDS development programs.

Approach: The organization, promotion, and execution of a successful technical conference is a major undertaking. The recommended approach would be to solicit joint sponsorship of such a conference by existing technical societies such as SME, AIAA, IEEE, etc.

Resources: The estimated resources for this project is one man year per year for the life of the IISEE program.

Timing: One conference per year.

Constraints: Coordination with the IDS and IISEE program offices and other societies technical conference activities.

Project Title: Managing an IDS Implementation Project

Project Goals: The goals of this research and development effort will be to identify and document the methods and tools required to effectively manage an IDS type of project. The long term targets will be to establish a comprehensive educational package with appropriate software for training project managers in the practices of IDS project management.

Project Objectives: The main objectives of this work will be to expand on existing project management tools and software to satisfy the needs of IDS projects. Several packages such as the CPM Based Project Managers Workbench and the Harvard Project Manager software will be explored for possible extensions into the management and control of IDS projects.

The characteristics, needs, and requirements of an IDS project will be defined and used as a baseline against which an educational package will be structured. Software to guide and support an IDS manager will be written to automate the IDS management function.

Background: Project management provides an organization with the tools required to improve the organization's ability to plan, implement, and control its activities and the way it utilizes scarce resources. When a project must be completed within critical time constraints and a clearly defined set of outcomes must be accomplished, an organized set of procedures and tools must be utilized to plan and implement such a program. The rationale behind the project is to provide the current project managers or the prospective IDS project managers with an educational package that will allow them to learn the basic techniques of scheduling and the allocating scarce resources to accomplish the goals of the IDS program. The package will be structured to allow the manager to learn through independent study or through an instructor lecture format. Case studies allowing the manager to gain an understanding of the problems encountered in an IDS project will be included in the package.

Approach: The approach taken will be to establish the needs of the IDS program manager within the context of the systems development life cycle. Interviews with individual responsible for the development of integrated information systems will be performed to determine through their experience the actual program management needs. Using the established needs as a baseline, existing project management approaches, software and concepts will be modified or developed to satisfy the program needs. After testing and structuring the tools into a package, a technology transfer manual for use by program managers will be written. The manual will contain slides for use in an instructor lecture format and narrative for use without an instructor. The manual and the narrative will be tested at actual facilities and case study scenarios will be developed during the refinement process. The educational package will consist of three separate volumes. They will be Volume 1 - text and narrative, Volume 2 - project management software, Volume 3 - case studies and scenarios. The three volumes will be coordinated with a roadmap type of structured overview document.

Resources: The development of this project management package will require two person-years of work. The individuals involved in the development of this project must have both project management and software development skills. Several software packages will have to be purchased and tested to determine the feasibility of their application in the context of the IDS project management framework.

Timing: Although the project will require two man-years of effort, the calendar time for the project should not exceed nine months to one year.

Constraints: The major constraints for this project will be locating people who are involved in IDS project management and soliciting their ideas. An enabling technology will be the extensions to existing software.

Chapter 4

Method Formalization

This chapter consists of an informal account of a formalization of the information modeling method known as IDEF₁ and an introduction to the concept of a neutral representation scheme as a basis for model integration. The IDEF₁ modeling method was developed around 1980 for the US Air Force to support aerospace manufacturers in developing an understanding of their manufacturing practices. Although widely used, it has never been subjected to rigorous formalization.

4.1 Rationale

Our attempts to develop rigorous formalizations of existing information modeling techniques have been motivated by the following observations and intuitions:

- Most current methods for information modeling have evolved as a distillation of “good practice” experience by information system developers. However the problems facing developers of large-scale integrated information systems are demanding the use of multiple techniques and the extension of existing techniques. Our intuition tells us that if we had a better understanding of:
 - what the current techniques can and can’t do,
 - why the current techniques do or don’t work,
 - how one technique relates to another technique,

then we should be able to “design” the interfaces between existing techniques, predict the extensions that are needed, and to design any needed technique extensions. The reason that this is such an attractive goal is that we would like to avoid the cost and pain associated with discovering voids in these methods through a series of failures in the systems we are currently attempting to build.

- The process of building integrated information systems is complex enough that many different models using different methods must be built. Our intuition is that if we can define the formal structures of these modeling methods, then we will be able to develop a more general representation for information extracted from various models that will ease translation and evolution from one modeling method to another.
- The systems we are trying to build (eg, IDS and IISS) require the direct manipulation of objects defined with information modeling methods. Without an unambiguous way of interpreting what those models mean we cannot guarantee that the systems will work the way we intend. We also find that an inordinate amount of guess work (gold dust as Bill Putnam calls it) is required on the part of design and implementation teams in attempting to interpret the models that are being produced.
- If we are to build the number of systems required of the complexities currently being contemplated we will need to train a large number of capable practitioners in a short period of time. The lack of rigorous formalization makes it difficult to train new people in the use of the methods and also makes it difficult to ascertain the competence of those who have been trained.
- The syntax of most of the current methods were designed for use by information systems personnel. Experience has shown that the information needed to build the models must be acquired from domain experts, and only the domain experts can validate the models. If through the formalization process we can separate the syntax of a method from the information that syntax is meant to display, then we can design alternative syntaxes which capture and display the same information in a form more palatable to the managers, engineers, and manufacturing domain experts.
- One of the stumbling blocks in the construction of automated tools for support of existing methods is that without a formal definition of "what the symbols mean" it is difficult to implement anything more complex than drafting and dictionary (text storage) tools.
- In order to be able to construct mechanisms which support the integration of methods, and to provide automated support for the transition between needs, requirements and designs, there is a need for a deeper understanding and representation of the semantics of the models constructed using existing methods.
- Without a rigorous definition of methods there is no way clearly to compare and contrast various techniques. This makes it very difficult for companies attempting to perform strategic planning, tactical planning and design to choose a technique. Minor syntactic changes can be used to hide concept identity, and major conceptual inconsistencies can be introduced with informal notions that "seem" intuitively correct. The

demand of the current generation of potential users is: *No representation without denotation.*

4.2 Approach

The course we have charted involves an attempt to develop a formal account of the syntax and semantics of four information modeling/database design methods which include: IDEF₁, IDEF1X, ENALIM, and ER. Our choice to begin with IDEF₁ is based on several reasons. First, it is a well conceived, well tested modeling technique that has been applied effectively in numerous situations; second, the authors of this chapter knew they would in the course of their work be able to meet several times at length with Timothy Ramey, the chief architect of IDEF₁. It is thus to IDEF1X that we will turn our attentions after the completion of our formalization of IDEF₁.

The process for formalizing an existing methodology turns out to be quite arduous and experimental in nature, consisting of the following activities (more or less performed in sequence with a lot of looping back):

- Examine existing documentation for the technique in question and begin extracting its underlying syntax and intended semantics. (If possible, this first stage involves the participation of the original developer(s) of the technique.)
- Attempt to formalize the syntax and semantics. This involves the choice of a lexicon and the development of formal grammatical rules, and the design of an appropriate mathematical interpretation of the syntax in terms of functions, relations, sets, and other mathematical objects.
- Identify problems both within the formalization (e.g., possible misinterpretations of the technique), as well as apparent problems uncovered within the technique being formalized (e.g., identification of redundant or inconsistent concepts in the technique).
- Work closely with the original developers (i) to clear up misinterpretations, (ii) to introduce them to the formalization as far as it has been developed, and (iii) to gain their assistance and guidance in continuing the formal development.
- Show that each well-formed syntactical "model" generated by the grammar has an interpretation, i.e., a set theoretic representation of a possible real-world modeling situation. This ensures that the grammatical rules are sound.
- Show that each interpretation can be represented in the syntax. This ensures that the grammar can represent any possible modeling situation which it is designed to capture.
- Suggest possible extensions to the modeling technique in question.

- Begin work on formal comparisons with previously formalized modeling techniques.
- Use insights gained to add to continuing work on the more general modeling technique.
- Move on to the next method.

The result of this process is the formalisms themselves and a documentation of the problems uncovered with the “perceived” meanings of models and rules for constructing those models, and the actual meanings and intended rules.

4.3 Informal Presentation

In this section, we will talk through the formal syntax of IDEF₁ along with the intuitive and informal semantics that informs and motivates the syntax.

4.3.1 The Syntax and Its Informal Semantics

Formally, IDEF₁ is treated as a symbolic system with two components: a precisely defined, but wholly uninterpreted syntax, and a set theoretic semantics. In this respect, our approach resembles exactly the standard treatment of first-order languages and their semantics in the branch of mathematical logic known as model theory. (Though, it should be emphasized, it would be inaccurate to think of IDEF₁ as a first order language, or of its semantics as a first order semantics.) The syntactic component itself consists of two components: a *lexicon* and a *grammar*. The lexicon for the IDEF₁ formalism consists of the following classes of symbols: upper case roman A's with numerical subscripts, which we call *entity class names*; lower case roman a's with numerical subscripts, which we call *owned attribute names*; and upper case roman L's with numerical subscripts and any of three superscripts—an arrow, a bullet, or a diamond—which we call *link names*. In practice, of course, instead of single letters one would use words intended to suggest the actual semantics of the names in the given situation.

Intuitively (and informally), entity class names denote classes of what we might call “the information image of objects,” (or conceptualized objects) rather than just “objects”. (Note carefully that we are talking here about *names* on the one hand, and the things they *denote* on the other. Entity class *names* denote entity classes, but entity classes themselves, being nonlinguistic sorts of things, don't denote anything. This crucial distinction was found to be missing or confused in the documentation of all modeling methods reviewed to date.) The reason we talk about conceptualized objects is that in concept modeling and data base design it is not typically real world objects themselves that are of interest (we can't write an employee on a hard disk), but rather certain clusters of information *associated* with real world objects relative to their (perhaps manifold) roles within a given system. The different roles that an object can play is reflected in the fact that the pertinent

information about that object conceptualized in one way (e.g., as a mere employee) will differ from the pertinent information about the object conceptualized in another way (e.g., as a corporate executive). The object thus has several distinct "projections" or "personae" in the informational structure of the system corresponding to these different ways of conceptualizing it. These projections are the conceptualized objects, hence it is these that are the members of entity classes.¹ Since entity classes are just classes of similarly conceptualized objects, and since there is a distinct conceptualized object for each way of conceptualizing a given real world object in an actual system, it follows that we cannot have the same conceptualized object occurring in more than one entity class.

The differences between entity classes is reflected in the *attribute classes* associated with the entity classes: distinct entity classes are necessarily associated with distinct sets of attribute classes; indeed, entity classes are essentially *defined* by their associated attribute classes. We thus need to take a closer look at the notion of an attribute class. Attribute classes come in two varieties: owned and inherited. Intuitively, owned attribute class names, as the astute reader will have deduced, denote the former. Owned attribute classes are simply those attribute classes that are uniquely descriptive of the members of a given entity class, and are not in any sense "derived from" (about which more shortly) any other entity class. Obvious examples here are attribute classes like *emp#*, and perhaps also *typing speed*. In the formal semantics of IDEF₁, attribute classes are *functions* in the mathematical sense.² That is, an attribute maps each member of a given entity class to a unique value. Thus, *typing speed* might be an attribute owned by the entity class SECRETARY, which would map the elements of that class into positive integers representing the number of words per minute each secretary can type.

Inherited attributes are another matter entirely. To get at the difference, we need the notion of a *link*.³ One of the most distinctive features of an information system is that the entities in the system are all interrelated; none of the conceptualized objects is an island. More exactly, it is entire entity classes that are interrelated. We can parse this idea in terms of links between entity classes. Links too are best thought of as functions, though, unlike

¹It is worth mentioning the philosophical connection between the ideas here and the distinction between sense and reference introduced by Gottlob Frege, the great German philosopher and inventor of modern logic. Just as an object can have many names each with its own sense, depending on how the name was introduced, so an object can have many distinct personae depending on the roles it plays in the system.

²Given this view it is confusing to use the term "class". Therefore in the remainder of this chapter we will use the term *attribute* to refer to what Ramey calls *attribute classes*. It should be noted that in the informal intuition associated with the IDEF₁ the distinction is very important since one of the points that Ramey was attempting to clarify is the need to build "class models". We believe that is a unique and important contribution which has unfortunately been overlooked by several more recent developments.

³What we're calling links actually correspond more or less to what Ramey calls *link classes*. Given that link names correspond semantically to functions, Ramey's term seemed to us inappropriate. What Ramey calls links, i.e., particular connections between two elements of two entity classes, we can call something else, e.g., connections, or perhaps link instances. As it happens, this is a notion that is unneeded in the formal development of the theory.

attributes, ones that map each element of an entity class to a single element of *another* (possibly the same) entity class. Thus, the link *works_for* can be thought to map each member of *EMPLOYEE* to a unique member of *DEPARTMENT*. Note that link functions can be composed to form "longer" links. Thus, *is_part_of* might be a link that maps *DEPARTMENT* into *DIVISION*, and hence the composition $(\text{is_part_of}) \cdot (\text{works_for})$ maps *EMPLOYEE* into *DIVISION*. An inherited attribute can now be understood straightforwardly as a composition of some number of link functions and an owned attribute. Thus, if we suppose that *div#* is owned by *DIVISION*, the composition of *div#* with the composite link above yields an inherited attribute of the members of *EMPLOYEE* that takes each member to his or her division number.

With this example it will be easy to show how inherited attributes are represented in our grammar. Let $a_3 = \text{div\#}$, $L_1^\circ = \text{works_for}$, and $L_2^\circ = \text{is_part_of}$. The composite link from *EMPLOYEE* to *DIVISION* is thus depicted as $L_1^\circ \cdot L_2^\circ$, and the full-blown inherited attribute as $L_1^\circ \cdot L_2^\circ \cdot a_3$. Such strings of link names, raised dots, and an owned attribute name are called *inherited attribute names*.

Now, as noted above, each role, hence each entity class, is determined by an associated bunch of attributes. Thus, in our grammar, we will group together entity class names with attribute names (of either sort) to form what we will call *entity class schemes*. We need to fit another piece into the puzzle first, however. One thing that is required in any classification of entities in a system is that there be some tangible way of distinguishing any particular element of a given class from any other. More specifically, for any entity class, there must be some nonredundant (possibly single-membered) collection of attributes chosen from among those associated with the entity class that jointly serve to identify and distinguish the elements of the class.⁴ Any such collection will be called a *key class* for that entity class. In our grammar, key classes will be represented by strings of attribute names enclosed in parentheses; such parenthetical strings will be called (to no one's surprise) *key class names*.⁵

We now have four different syntactic items having to do with entity classes: entity class names, owned attribute names, inherited attribute names, and key class names. These four we combine into complex expressions that we call *entity class schemes*. Intuitively, entity class schemes explicitly display the full logical structure of an entity class. We construct them in the following way. The idea here is to squash the usual IDEF₁ boxes with their outgoing arrows flat so as to make them more amenable to formal treatment.

First, we can think of beginning with a template that looks like this:

$$[(\dots)(\dots)(\dots)]$$

⁴Such a collection is "nonredundant" if it is not possible to remove any of its members and still have a collection that uniquely distinguishes the elements of the entity class.

⁵Though not just any such string will count as a key class name. For various reasons we will not go into here, for example, one cannot have a key class name containing two inherited attribute names such that the leftmost link name in one is many-one, and the leftmost link name in the other is one-one. This has to do with important facts about how one-one links in information systems function.

Then we stick an entity class name out front. Next we fill in the spaces between the angle brackets with some key class names between the first two, some inherited attribute names between the second two, and some owned attribute names between the last two in the following way. Let ϵ stand for any arbitrary entity class name, $\kappa_1, \dots, \kappa_i$ any key class names, ι_1, \dots, ι_j any inherited attribute names, and $\omega_1, \dots, \omega_k$ any owned attribute names subject to the following restrictions: (i) none of the key class names can contain all the attributes occurring in one of the other key class names; (ii) every one of the inherited attribute names whose leftmost link name is one-one must occur in one of the key class names; and (iii) every owned (inherited) attribute name occurring in any of the key class names must appear among all the owned (inherited) attribute names between the angle brackets. (This latter requirement may appear redundant, but it turns out to be very convenient for formal purposes.) Then the general form of an entity class scheme is

$$\epsilon[(\kappa_1, \dots, \kappa_n)(\iota_1, \dots, \iota_j)(\omega_1, \dots, \omega_k)]$$

6

A concrete example would be this. Let $A_1 = \text{EMPLOYEE}$, $a_1 = \text{emp\#}$, $a_2 = \text{dept\#}$, and as above, let $a_3 = \text{div\#}$, $L_1^* = \text{works_for}$, and $L_2^* = \text{is_part_of}$. Then, supposing first that emp\# , dept\# , and div\# are the only attributes that distinguish the entity class **EMPLOYEE**, and that each element of the class is identified jointly by its emp\# and its div\# our entity class scheme would look like this:

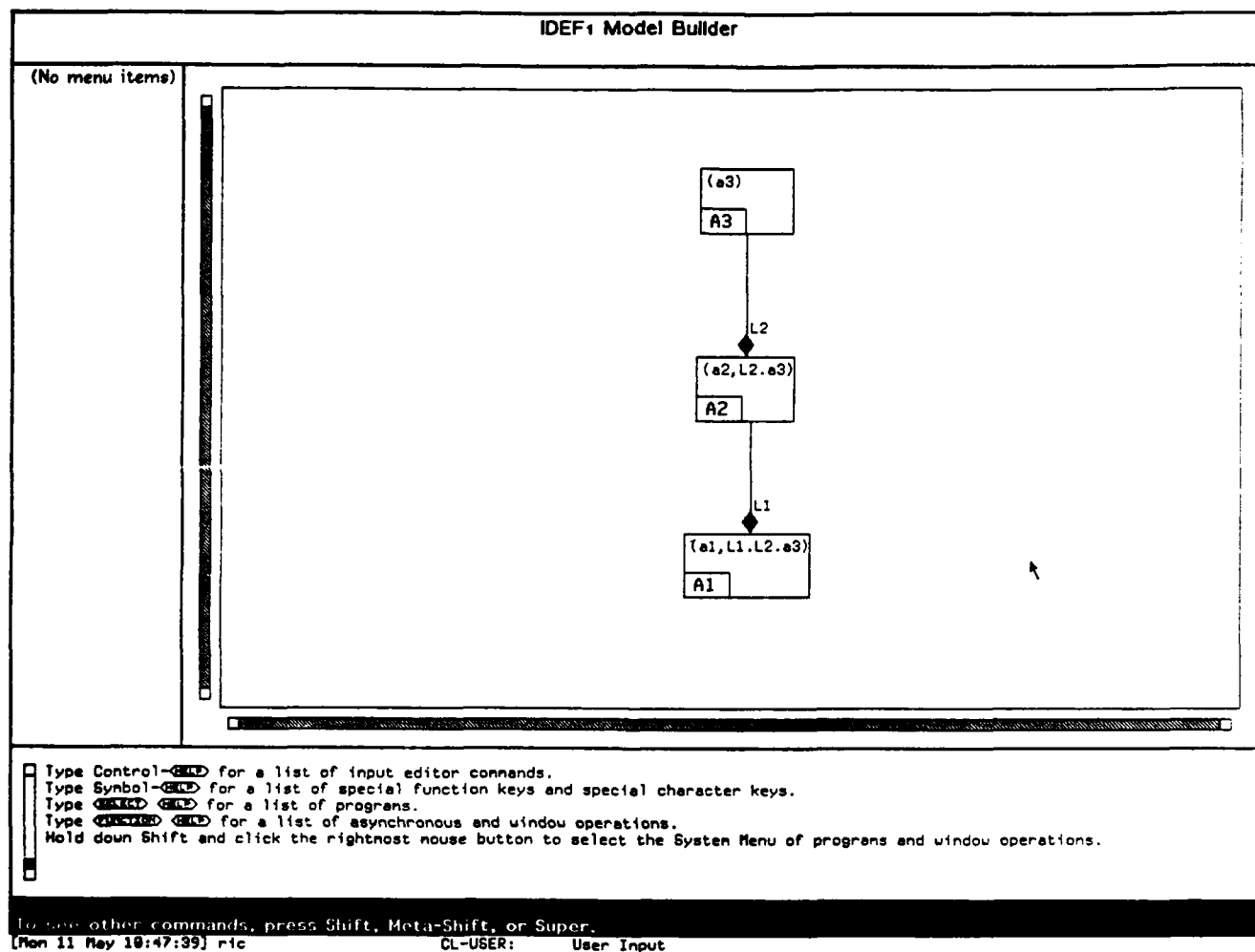
$$A_1[(a_1, L_1^* \cdot L_2^* \cdot a_3)(L_1^* \cdot L_2^* \cdot a_3, L_1^* \cdot a_2)(a_1)].$$

If we let $A_2 = \text{DEPARTMENT}$ and $A_3 = \text{DIVISION}$, the entire situation between these three entity classes that we've envisioned can be represented as in the more familiar looking graphical IDEF₁ diagram seen in Figure 4.1.⁷

Once we have our legal entity class schemes in hand, the grammar tells us how to build proper IDEF₁ diagrams out of them. Specifically, the grammar defines one entity class scheme Σ to be *linked* to another Σ' via some link name λ just in case λ is the leftmost constituent of some inherited attribute name ι occurring in Σ , and the attribute name that results from deleting λ from ι occurs in Σ' . In Figure 4.1, the entity class scheme A_2 (more exactly, the entity class scheme labeled 'A2') is linked to A_3 via L_2 , and the entity class scheme A_1 is linked to A_2 via L_1 . The idea of one entity class name being linked to another

⁶In constructing a formal system of any kind one must use a *metalanguage* to talk about the system one is constructing. Often in such a process, one wishes to talk about the members of an entire *class* of syntactic items generally, and not just one or two in particular from a given class. An example is our use of ϵ in the general form of an entity class scheme here to talk about *all* entity class names, and not just about, say, A_1 . ϵ is thus an example of a *metavariable*. It is important to realize that these greek letters are not themselves part of the language but are being used only to talk in general terms about the elements of the language, just as one might (as in fact we have) use the letters i, j , and k to talk in general about the natural numbers, or A, B and C to talk in general about geometrical lines.

⁷The diagram in Figure 4.1, drawn by a program that enables one to create IDEF₁ diagrams on a Symbolics Lisp machine, actually uses the correct notation for strong many-one link names, viz., a solid diamond. It was easiest for the purpose of this report simply to use bullets in the text. Note also that in actual practice, where there is no danger of ambiguity, the link names in an inherited attribute name can be suppressed.

Figure 4.1: Sample IDEF₁ Diagram

reflects syntactically the idea of one entity class inheriting an attribute from another via some actual link between them.

Using this notion, we then define what amounts to the usual graph theoretic notions of a *path* and a *walk* in a set S of entity class schemes. A path in S is defined to be *increasing* (*decreasing*) if each entity class scheme in the path—save the last—is linked to its successor via a one-one (strong many-one) link name, and *cyclic* if the first and last schemes in the path are identical. A walk in S is defined to be increasing just in case the result of changing all the strong many-one link names in the walk to one-one link names is an increasing path, and, as with paths, it is defined to be cyclic if the first and last schemes in the walk are identical. We then define S to be *connected* just in case there is a walk from any entity class scheme in S to any other. Thinking semantically, the idea here is that in a correct account of an information system, one should be able to find a route from any given entity class to any other by traversing either forward or backward along the links between entity classes in the system. The natural way to express this syntactically (as we shall do below) is with the requirement that the set of entity class schemes representing the system be connected.

We can illustrate these concepts pictorially by extending Figure 4.1 to Figure 4.2. In Figure 4.2, there is, among others, a path from A4 to A3 along the “links” (more correctly, link names) L3 and L2, and one from A1 to A4 along the links L1, L2 and L4. While there is no path from A4 to A1, there is a walk, in fact many of them. One can traverse forward on the link L3 and backward on the link L1, for example, or backward on L4, L2, and L1. There is an increasing path from A3 to A4 along L4, and a decreasing one from A1 to A3 along L1 and L2. Finally, there are several cyclic paths, e.g., the one from A2 to A2 along L2, L4, and L3. The diagram is obviously connected, as there is a walk between any two entity class schemes.⁸

Given these definitions, we define a set S of entity class schemes to be a *prediagram* if it is finite, if no distinct entity class schemes share the same entity class names or any of the same attribute names, if no more than two entity classes are linked via any given link name, and if every member of S is such that if it contains an inherited attribute name ι , then it has to be linked to exactly one other entity class scheme in S via the leftmost link name in ι . The intuition here is that we can't have any “dangling” link names that are attached to an entity class scheme at only one end; semantically put, if an entity class includes an inherited attribute among its distinguishing attributes, then quite obviously there has to be some entity class from which it is inherited.

In order for a prediagram S to qualify as a full-fledged IDEF₁ diagram, we require (in addition to a couple of restrictions on inheritance we'll not bother with here) that no cyclic walk be increasing. The reason for this is that the only way that such a walk could possibly represent an actual information system, say, would be if all the links in the walk between entity classes were always *one-one* and *onto*. But by the nature of one-one links, entity

⁸Figure 4.2, however, is not a legal diagram as it stands, since no key class name is inherited by A3 through L4.

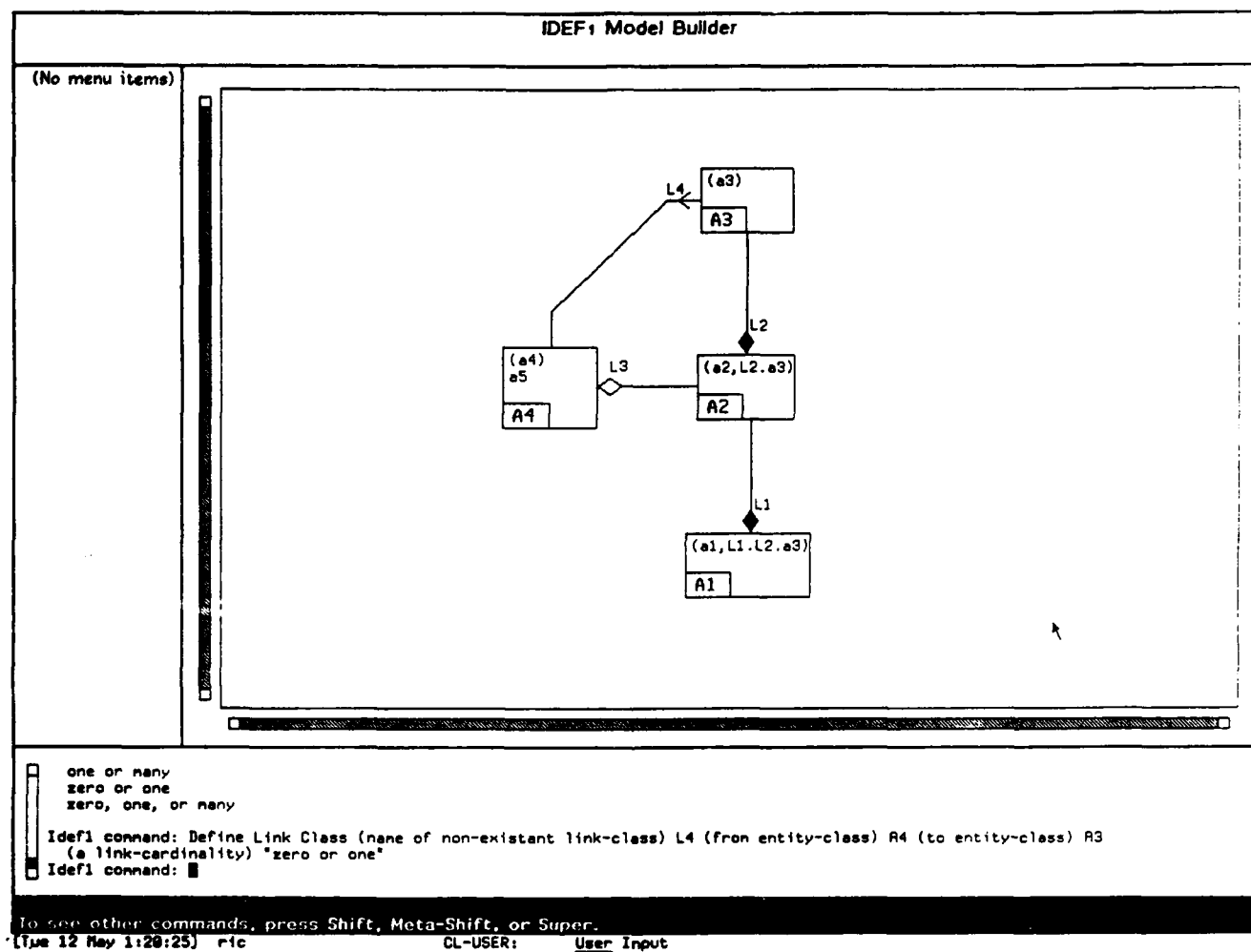


Figure 4.2: Paths and Walks in an IDEF₁ Diagram

classes so linked would have to be "informationally synonymous," and hence one of them would be superfluous. As a rule, the identification of such a link in the analysis of a system indicates the presence of an overlooked attribute.

4.3.2 Formal Semantics

The discussion above indicates pretty clearly how the formal, set theoretic semantics for the formal syntax works. The basic moves go like this. We first provide interpretations for the elements of the lexicon. We start with a finite set of mutually disjoint finite sets. Intuitively, these represent entity classes, and are the semantic values of entity class names. Link names are assigned functions from one entity class to another (possibly the same) entity class. Next we have another set of sets, intended to represent owned attribute values. Each owned attribute name is thus assigned a function from an entity class to one of these latter sets. The semantic value of an inherited attribute name is then defined recursively as the composition of the semantic values of the attribute names (from right to left) that make it up.

Next we say what it is for such an interpretation of our lexicon to *satisfy* an entity class scheme. This will be the case, essentially, if the functions assigned to its attribute names take the class C assigned to its entity class name as their domain, and if the functions in each key class jointly (and nonredundantly) identify and distinguish the members of C . An interpretation can then be said to satisfy a set of entity class schemes, and in particular, an IDEF₁ diagram, just in case it satisfies every member of the set.

4.3.3 Metatheory: Consistency and Completeness

Once we've defined the formal syntax and semantics for IDEF₁ diagrams there are a few things we'd like to know about what we've defined. Loosely put, we want to know that the syntax and the semantics "hook up" in the right sort of way. Somewhat more precisely, we want to know, first, that our syntax is in a certain sense *consistent*. That is, we want to know that the rules of the grammar are good ones, in the sense that they only allow us to construct "coherent" IDEF₁ diagrams, i.e., diagrams that could be used to characterize some (perhaps very bizarre) possible information system. To prove this, one has to show that every IDEF₁ diagram that the grammar generates in fact has an interpretation, in the formal semantical sense. At present, a strong partial result has been proved for all noncyclic IDEF₁ diagrams, and for certain cyclic diagrams that are not in a certain sense too "complex". Early indications are that this result will generalize quite straightforwardly to all possible IDEF₁ diagrams.

The second thing we want to know is that our grammar is in a certain sense *complete*. That is to say, we want to know as well that the rules of the grammar are strong enough to be able to capture any possible information system that we can construct in our formal semantics. This is quite easy to prove, given the rather elaborate and specialized nature of

the syntax; for given some formal information system (as defined in the semantics) one is able to construct an IDEF₁ diagram straightaway simply by reading off the structure of such a system directly.

It is important to realize that, with these proofs in hand, one is *guaranteed* that the IDEF₁ modeling technique is sound—that its grammatical rules won't permit senseless diagrams, and that, relative to its conception of the structure of information, there is no possible situation it cannot represent. But notice that these results are possible only within the context of a rigorous, mathematically precise formalization of the technique's syntax and semantics. It is our view that a minimum requirement on any modeling technique should be that it be susceptible to similar formalization, and that it be provably consistent and complete in the senses developed here.

One of our chief goals is to establish that this requirement can be met by all the modeling techniques mentioned at the beginning of this chapter. Once established, we will then be in a position to compare and contrast the various techniques with regard to their views of the structure of information, their relative expressive power, and so on. And perhaps most important, we will possess the necessary theoretical foundations for the development of a neutral information representation scheme (NIRS).

4.4 Neutral Information Representation Language

This section describes the concepts, rationale, and preliminary requirements for such a representation scheme referred to as the "Neutral Information Representation Language" (NIRL). In concept the NIRL is intended to be a computer processable medium for representing the results of application of the component methods associated with an ISEE framework. It might better be thought of as a knowledge representation language that is so designed as to be able to represent the knowledge about both the "AS IS" system and the evolving "TO BE" system. Its primary purpose is to support the movement of information

1. Between methods in a particular methodology (e.g. from ENALIM to IDEF1-X).
2. Between methods in different methodologies (e.g. from IDEF0 to IDEF1-X).
3. Between methods used in different phases of the life cycle (e.g. from an IDEF1-X to a Data Structure Diagram).

The endgame envisioned for the NIRL is illustrated in Figure 4.3. Once we can isolate the information gathered and the information displayed by a method then we want to be able to store that information in such a manner that it can be used in other stages of the life cycle, or displayed in alternate forms. That is, it was the experience of the coalition that as one moves from one phase of the development process to the next much of the information which is gathered and displayed in one form is needed in the next. It was also the experience of the coalition members that there are personal and organizational preferences for particular

methods within a methodology. What we would like to be able to support with a knowledge representation system designed around the NIRL is the ability to perform deep structure translations between the methods.

The process of definition of the NIRL can be thought of as a collecting together of the semantic structures uncovered through the formalization process described in the previous section. These concepts then must be analyzed and a set of structures defined for representation within the ISEE environment.

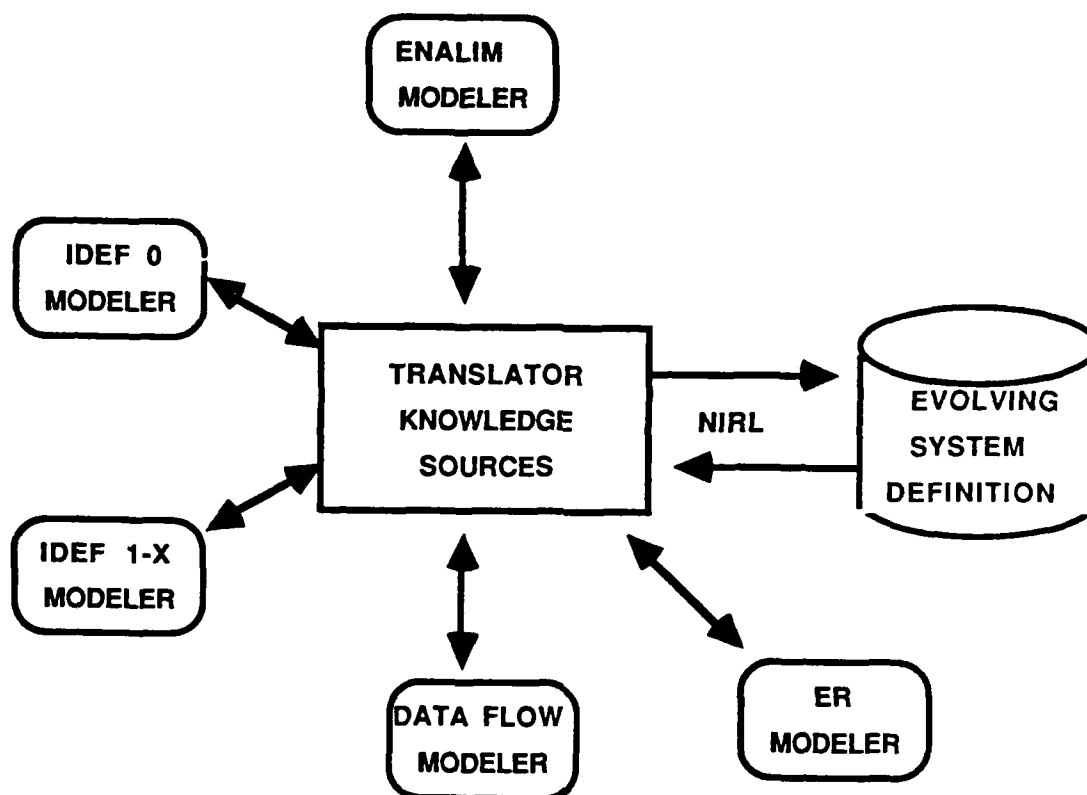


Figure 4.3: NIRL Support for Methodology Integration

Chapter 5

Natural Language Processing of Manufacturing Texts

The motivation for studying the application of natural language processing (NLP) on information models such as IDEF and the texts and glossaries accompanying them lies in the practical problems with building models (model generation) and with understanding them (model interpretation). In many instances, no model is used at all because of the difficulties of model construction and the time required by that activity. In other cases, models that are constructed are of low quality or are not uniform. Moreover, once a model is produced, users often have trouble understanding it. The lack of compatability between different types of models provides a further hindrance to comprehension. The work with NLP confronts these problems by examining the feasibility of model generation from English sentences (to assure high quality and uniformity), the feasibility of interpreting the information content of the models in English sentences, and the feasibility of using NLP as a device for model translation using the NIRL developed as part of this project.

To explore the feasibility of NLP, we have undertaken four activities, which include:

1. Conducting a survey of NLP approaches to determine which would be most useful for our particular application.
2. Performing a linguistic analysis of a set of manufacturing texts to determine whether or not NLP was possible. These texts included IDEF models, glossaries, and texts from the *System Environment Document, Integrated Sheet Metal Center, Volume II, 15 April 1982, Boeing Military Aircraft Company* and a list of business rules from the *Integrated Design Support System (IDS) Technology Review No. 3, August 1986* and free-format manufacturing text collected from various manufacturing companies. Sample texts are included as Appendix A.
3. Prototyping a system that would process the model statements from the Boeing Integrated Sheet Metal Center System Document and the business rules from the IDS

Technology Review.

4. Determining the most feasible kinds of applications for NLP.

5.1 Approaches to Natural Language Processing

Developing a natural language processing system (NLPS) is an extremely difficult task, although it might at first glance seem deceptively easy because of the ease with which humans, including small children, manipulate and understand language. Understanding natural language involves not only understanding the meaning of individual words but also understanding the meaning and function of those words within a sentence or even a body of discourse. However, many English words are ambiguous; that is, the same word can be used either as different parts of speech or with different meanings. Further, in developing an applications-oriented NLPS, we have the classic problems of linguistic theory, including anaphora, elipsis, and coordinate conjunctions, that remain open research questions, as well as the inherent ambiguity of English syntax. Humans process ambiguity because of a background or world knowledge that sets the context of the discourse and builds a script of what is expected. It is the difficulty of building into a NLPS this background knowledge component and a method for recognizing the functions of words in blocks of discourse that makes the task of designing computer NLPS so difficult. Our emphasis within this work has been to constrain the difficulty in developing a NLPS in order to make a practical application.

A NLP consists of a parser and a semantic interpreter. The parser determines the syntactic structure of a sentence while the interpreter assigns meanings to these structures. To assign words to syntactic categories or parts of speech, the parser makes reference to a lexicon. Once assigned a part of speech, these lexical items are then assigned to larger syntactic structures such as noun, verb, or prepositional phrases according to a grammar component.

In its simplest form, the lexicon is a dictionary of possible lexical items and their associated syntactic categories. However, such a simple lexicon has a serious drawback: many English words are inherently ambiguous on several different levels. First, a word may be syntactically ambiguous, with several possible category assignments. Thus the word *run* may be either a noun or verb; the word *that* may be a pronoun, determiner, or subordinator. Second, a word may be lexically ambiguous, with several different meanings associated with the same syntactic category, as in the word *run* which can be used as either a verb or a noun. As a verb, it has a range of meanings from "*run* a race" to "*run* away" to "*run* a risk" to "*run* a fever", while as a noun it can have such various meanings as in "a 10K *run*" "a long *run*" "the usual *run* of men", or "a *run* in my stockings". Humans easily process and understand these various senses because of the semantic cues provided by the larger body of discourse. Unfortunately, a large number of English words are ambiguous in one or both ways. However, because the language of the models is so restrictive, much of the lexical ambiguity is eliminated in our task, but the problem of syntactic ambiguity, as discussed

below, still remains.

The best strategy for actually parsing and understanding sentence components of text is an open research question. Of the many prevailing approaches to NLPS, most fall into two broad categories — linguistic (syntax driven) or conceptual (semantics driven). Linguistic systems maintain the parser and interpreter as separate components, but conceptual systems use some elements of the interpreter to constrain the parser. One such constraint mechanism is the use of case frames, that is the assignment of syntactic categories based on the semantic / syntactic relationship of constituents to the predicate of the sentence. Another is the use of a sublanguage system, or a lexicon constrained by a particular domain.

Linguistically based systems rely primarily on knowledge of grammar (the syntax and morphology of a particular language) rather than on knowledge of a particular domain. Because these systems are syntax-driven, they may actually generate several meanings for the same sentence and may in fact generate interpretations that make no sense in the real world. Conceptual parsers, on the other hand, are guided in their parsing by their knowledge of some domain and thus eliminate interpretations which have no meaning for a particular domain. However, since conceptual systems are domain-specific, they cannot easily be generalized without the regeneration of the knowledge component. The grammar in a linguistic parser usually consists of a set of phrase structure rules used to bundle syntactic categories into syntactic constituents, but in a conceptual parser the grammar forms a set of sentence patterns anticipated by the system. Conceptual parsers attempt to incorporate some of the human's ability to disambiguate by providing extensive knowledge about a limited domain, thus imposing severe constraints on the use and co-occurrence of certain lexical items. For instance, in a conceptual parser, the verb *snore* would require a human agent and an optional indicator of manner. These constraints are imposed by syntactic category restrictions on words in the lexicon and by case frames associated with verbs.

Regardless of the approach, parsers accept as input a subset of the text (we will assume a sentence by sentence parse) and match each lexical item (or word) with possible syntactic categories (or parts of speech). The string of syntactic categories are bundled into larger and larger syntactic constituents by the parser until the sentence can be processed, and the information content of the sentence incorporated into some representation strategy.

5.2 Conceptual Parsing Approach

In our work we have adopted the conceptual approach because we believe that successful, efficient text processing and understanding depends upon building a computer system that combines grammatical knowledge with the expectations and constraints that an analyst brings to understanding a model and its associated text and glossaries. Thus we reject the notion for this application that a parser can operate autonomously on syntax without regard to the domain of discourse. Our parser uses expectations of textual organization built into patterns based on the structure of the material as well as grammatical expectations or

case frames. For this reason, our system would not understand texts which lie outside that domain. The patterns for manufacturing texts would expect only data relating to a manufacturing process, and further, only the kinds of data required by an IDEF model (that is, information descriptions) in one case and business rules from the IDS Technology Review in the other. The patterns also anticipate either the verb and preposition or the verb and case frame co-occurrence structures that help establish case frames. Since these structures are complex, the parser must sometimes rely on a scale of probability for understanding the meanings of lexical items which have a wide range of meanings. Our parser, for instance, would attempt to understand *run* as referring to a job (as a verb) or perhaps to a completed job (as a noun); it would not, however, understand "*run* in a stocking" or other meanings of *run* outside of the manufacturing domain.

Our parser makes use of two other devices. First, we have tried to establish a grammar of the sublanguage associated with manufacturing. Thus, in the case of model descriptions, we have tried to determine through a manual analysis of relevant texts, the set of verbs commonly used to establish entity and attribute relationships, the constraints on the possible meanings of those verbs, the caseframes normally associated with those verbs, and the range of syntactic patterns which occur. We have completed a similar analysis for the business rules from the IDS Technology Review. Second, we have utilized a "minimal lexicon" or a pattern-based approach that does not require that all nouns and adjectives be in the lexicon for correct parsing. Parsing is accomplished on the basis of the grammatical and conceptual constraints, as coded into a case grammar, developed in the manual analysis.

In the restricted domain established by the IDEF model texts, for example, we have structured our set of patterns to expect text in the third person. For instance, we certainly do not expect imperative sentences. In fact, in the glossary definition, many sentences which "appear" to be imperative (lacking a subject and beginning with a verb) are simply definitions where the missing subject is assumed to be the entity class label as shown in Figure 5.2. In other cases (especially in the definitions of entities and attributes) both a subject and verb are missing: that is, a sentence fragment is used in the definition. The subject again is the entity class label, but in this instance the verb must be inferred by the reader. (Note that this is the typical practice in both commercial and scholarly lexicography. Definitions are given as sentence fragments and especially as complements of verbs. The head word supplies the missing subject, while the reader infers the verb.) Supplying missing subjects in the glossaries is a trivial parsing problem; the correct verb is usually a form of *be* (an *is a* relationship, for example) with entity classes as shown in Figure 5.2. With attribute classes, the problem is somewhat more complex requiring more study.

Further, these patterns are associated with processing case frames which anticipate certain types of information once the verb phrase is isolated. Since this information is often denoted by the co-occurrence of verbs and prepositions or verbs and syntactic functions (such as subject or direct object), our processor is tuned to search for the anticipated semantic casemarkers.

USED AT: BMAC	AUTHOR: S. Y. LEE PROJECT: ISMC	DATE: 2/25/82 REV:	STATUS: DRAFT	READER	DATE	CONTEXT
NOTES: 1 2 3 4 5 6 7 8 9 10						
<p>ENTITY CLASS NAME : COMMITMENT BOARD COMMITMENT</p> <p>ENTITY CLASS LABEL : COMMITMENT BOARD COMMITMENT</p> <p>ENTITY CLASS DEFINITION: The synthesized agreement of involved departments to the assigned tasks and schedules in support of a program or contract.</p> <p>ENTITY CLASS SYNONYMS :</p>						
MODE: E526	TITLE: ENTITY CLASS DEFINITION: COMMITMENT BOARD COMMITMENT			NUMBER: x x		

Figure 5.2: Sample Entity Class Definition Where Both Subject and Verb Must Be Inferred

5.3 Text Analysis

As a first attempt at establishing the IDEF sublanguage, we did a manual analysis of the approximately 2400 model statements in the Boeing Integrated Sheet Metal Center System Document, one of the most complete IDEF models available. The IDEF statements set up relationships between entity classes which correspond to the relationships depicted graphically on the models, and thus, provide an ideal test case for NLP on the IDEF models. These model statements also demonstrate the importance of the verb, which sets up entity relationships in the manufacturing sublanguage. Figure 5.3 provides a representative sample of the model statements which we analyzed manually. In these 2400 statements, we found that only 53 verbs occurred, and with these 53, a limited set of caseframes occurred. Among the common caseframes are *agent* (performer of actions), *object* (thing acted upon or defined), *locative* (the place where things occurred), *equative* (which equates two things), *temporal* (time when something occurs), *dative* (the thing or person to whom or for whom the action takes place), *manner* (how something is accomplished), and *source* (the basis of an action). Further, not all cases occur with all verbs. In fact, there are severe constraints on the distribution of cases and our parser relies heavily on these constraints. Caseframes can be identified by one of two means — the occurrence of certain prepositions or word order. The meaning associated with prepositions is totally dependent upon the verb with which they co-occur. The same preposition may signal different cases with different verbs. For example, with some verbs *in* signals a locative case, while with others it signals a temporal case. Further, *in* functions as a verb particle with the verb *result*. What disambiguates the uses of *in* is the co-occurrence with a particular verb. These findings confirm our hypothesis that information models make use of a highly restricted sublanguage and leads us to believe that at least some NLP is possible in model generation and interpretation. (The business rules from the IDS Technology Review provided a different kind of problem, but after a similar kind of manual analysis, we believe that NLP is also possible with these texts. While a wider range of syntactic structures occurs in the texts, they can be parsed through the mechanism of successive parsing. For example, discontinuities in the verb phrase can be resolved in a preprocessing stage, thus enabling an accurate parse of the sentence.

The linguistic analysis of these two kinds of texts also reveals several interesting parsing problems. First, there is the syntactic ambiguity of words such as *manufacturing*, *request*, *charge*, *drawing*, *support*, and *issue*. Each of these words can be used as both nouns and verbs or adjectives and verbs. This ambiguity creates insurmountable problems for purely syntactic parsing. In such instances, heuristic devices have to be used to disambiguate the sentences. Second, the analysis reveals that the key to parsing the text lies in locating the verb. In the IDEF models, the case frame structure associated with the verb is closely associated to the relationship between entities. These entities are in fact components of the case frames. This last finding is particularly useful in light of the structure of IDEF models, which capture the entity / relationships. In the business rules from the IDS Technology

**Representative Sentences which the Prototype
can Parse and Understand**

1. Rework orders are reworked according to withhold tag item.
2. Tool type conforms to tool specification.
3. Resource need identifies requirement for resource plan.
4. With hold tag item becomes certified part after rework.
5. WOP validation verifies the contents of work order packages.
6. RLSD Eng drawing satisfies end item requirement for drawing.

Figure 5.3: Representative Sentences Which the Prototype Can Understand

Review, however, the interaction of verbs and cardinality relationship (for example, *one and only one*; or *zero, one, or many*) provides the semantic core of the sentences.

5.4 Prototype System for IDEF Statements

Using the information from the text analysis, we developed a prototype parser which parses almost all of the 2400 model statements in the Boeing Integrated Sheet Metal Center System Document. For our prototype, we are utilizing the Language CraftTM, a product of Carnegie Group Inc. because it utilizes the caseframe grammar which forms the basis of our linguistic analysis.

5.5 Prototype System for Business Rules

A second prototype system was developed for the business rules from the IDS Technology Review. Again, Language CraftTM was utilized for the prototyping. These business rules establish cardinality or hierarchical constraints using grammatical constructs such as *is a type of*, *is for exactly one*, *has zero, one, or many*, *has exactly one*.

5.6 Future Work

Using these prototypes as a basis, we are pursuing four other activities. First, it would be beneficial to implement an "Automated Model Generator" (AMG) which would provide a "workbench" capability for building IDEF models. Such a tool would have the advantage of making models easy to draw and of providing uniformity and consistency checking. Starting with statements similar to the statements in the Boeing Integrated Sheet Metal Center System Document, the AMG would produce graphic representations of entity relationships automatically.

A second activity would develop an "Automated Model Interpreter" (AMI) which would generate all English sentences which explain the implications of the IDEF models. This tool would have the advantage of making explicit the consequences of a model and would be significant step forward in providing the user an understanding of the models. The AMI would begin with graphic representations and have as its output English sentences.

A third activity, much further in the future, would interpret the glossary and texts associated with IDEF1 models. This is a far more complex application unless some restrictions are placed on the syntax used in text and glossaries. Essentially, four problems present themselves. The first problem is a "pseudo" problem of processing the sentence fragment structure. In fact, the subject of these sentences can be inferred from entity labels and the verb is generally a form of *be*. The other problems are more complex. For one thing, the glossaries contain a very broad range of syntactic structures including embedded sentences. The

texts themselves contain a much larger, unconstrained set of verbs and relationships as well as a broad range of syntactic structures. Both of these problems require an extremely robust grammar, a significant lexicon, and an extensive set of heuristics to account for potential ambiguities. Finally, the last problem deals with the complex set of relationships between attribute types and entities. Here parsing will require extensive background knowledge of a particular manufacturing domain.

The fourth activity involves providing information for the development of the NIRS (Neutral Information Representation System). This activity requires the establishment of canonical sets of verbs, entities, and relationships. It is the most far-reaching of the four activities and the one with the most general set of applications. We have already begun work on establishing canonical sets of verbs. For example, in the Boeing Integrated Sheet Metal Center System Document, we have been able to classify the 53 verbs into 12 categories based on the relationships they depict. Further work may enable us to classify these into even fewer categories.

APPENDIX A

Sample Text

USED, AT:	AUTHOR: S. Y. LEE	DATE: 2/18/82	STATUS:	READER	DATE	CONTEXT
BMAC	PROJECT: ISMC	REV:	DRAFT			
NOTES: 1 2 3 4 5 6 7 8 9 10						
568	<>--- 503 SHOP OPERATION --- 510 SHOP PERFORMANCE INDEX --- 557 WORK ORDER PACKAGE PARTS LIST ITEM --- 513 LOT TIME ITEM --- 296 PART LIST ITEM SPEC CALLOUT --- 556 P/L ITEM MATL CALLOUT --- 512 UNIT TIME ASSEM --- 578 PARTS LIST --- 6 PART --- 415 PROCESS PLAN	HAS RESULTS IN RESULTS IN IS SPECIFIES HAS REFERENCES IS MADE UP OF SPECIFIES RESULTS IN				
578	PARTS LIST --- 12 RLSD ENG DRAWING --- 568 PARTS LIST ITEM --- 542 PICK LIST --- 501 PLANNING COORD RECORD	HAS IS MADE UP OF IS USED AS HAS				
588	WORK ORDER VARIABLE --- 6 PART --- 557 WORK ORDER PACKAGE --- 93 FAB LOT --- 409 SCHEDULED PART --- 551 EOR LOT --- 552 MIN LOT --- 553 MAX LOT --- 554 FIXED LOT --- 598 WOP VALIDATION --- 407 AVAILABILITY OF MATERIAL --- 513 LOT TIME ITEM --- 519 ACTION ITEM CARD	IS REQUESTED BY RESULTS IN DEFINES ACCUMULATES THE REQUIREMENTS OF IS IS IS IS HAS AFFECTS THE RELEASE OF HAS EXPEDITES THE RELEASE OF				
598	WOP VALIDATION --- 588 WORK ORDER VARIABLE --- 557 WORK ORDER PACKAGE	HAS VERIFIES THE CONTENTS OF				
599	WOP/ COST ACCOUNT --- 4 COST ACCOUNT --- 557 WORK ORDER PACKAGE	IS CHARGED FOR HAS				
NODE:		TITLE: RELATED ENTITY CLASS NODE CROSS REFERENCE		NUMBER:		

US20 A1:	AUTHOR: S. Y. LEE PROJECT: ISMC	DATE: 2/10/82 REV:	STATUS: DRAFT	REVIEW	DATE	CONTEXT:
NOTES: 1 2 3 4 5 6 7 8 9 10						

61
EQUIPMENT

IS CURRENTLY AVAILABLE AT

116
DEPARTMENT/SHOP

CONTAINS

240
EQPMNT LOCATION

18

276
CUSTOMER EQPMNT SHOP ASSIGN

277
OWNED EQPMNT SHOP ASSIGNMENT

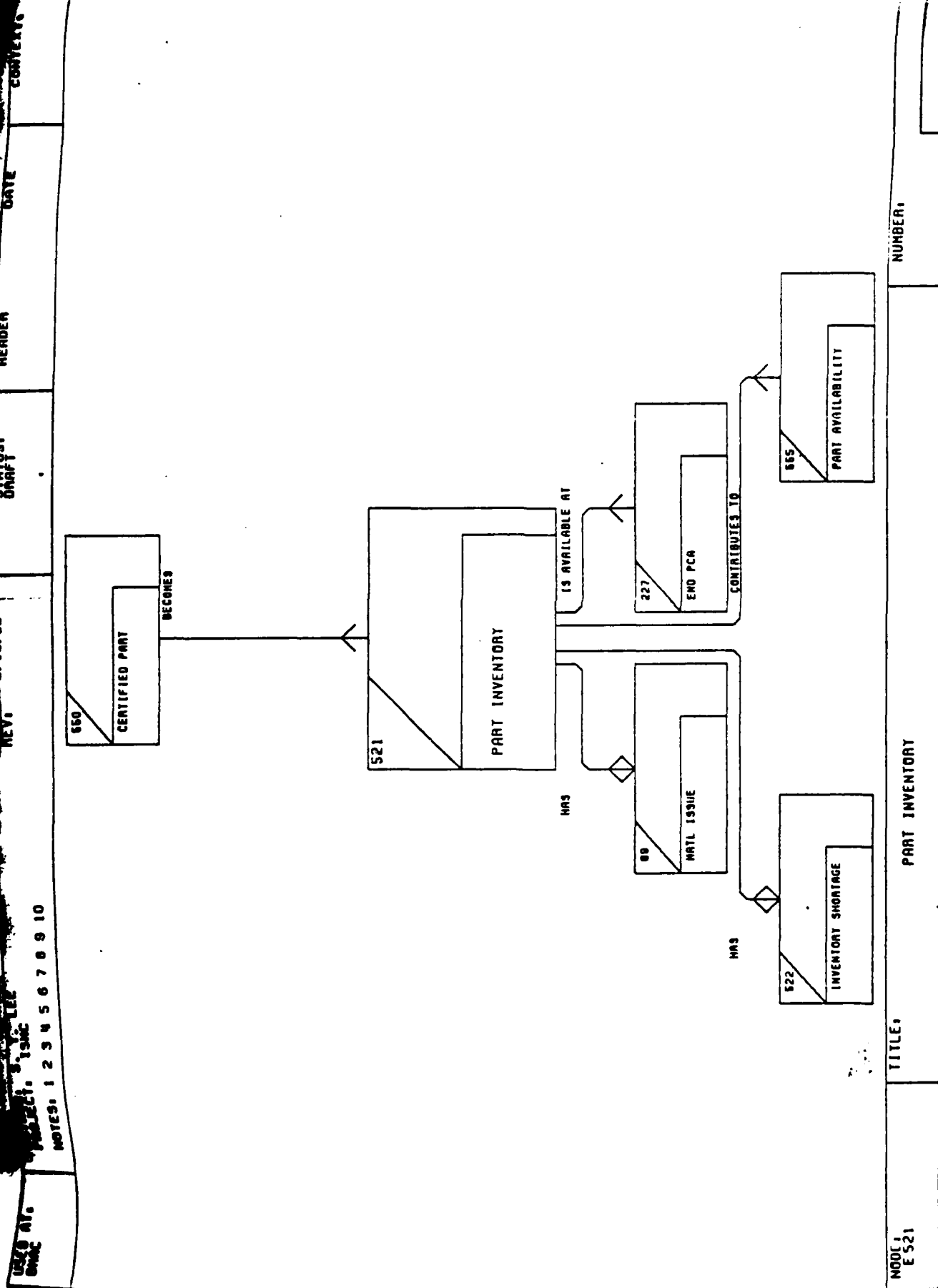
18

MODEL:
E 240

TITLE:

EQPMNT LOCATION

NUMBER:



USED AT: BMAC	AUTHOR: S. Y. LEE PROJECT: ISMC	DATE: 2/18/82 REV:	STATUS: DRAFT	READER	DATE	CONTEXT
	NOTES: 1 2 3 4 5 6 7 8 9 10					
	<>--- 115 DEPARTMENT/SHOP					
	<>--- 359 ENG ORDER USE FOR ENG CHANGE REQUEST					ISSUES
	<>--- 393 CHANGE REQUEST DRAWING CALLOUT					HAS
358	ENG ORDER USE FOR ENG CHANGE PROPOSAL					
	<>--- 332 ENGINEERING CHANGE PROPOSAL					RESULTS IN
	<>--- 361 APPROVED ENGINEERING ORDER					HAS
359	ENG ORDER USE FOR ENG CHANGE REQUEST					
	<>--- 354 ENGINEERING CHANGE REQUEST					RESULTS IN
	<>--- 361 APPROVED ENGINEERING ORDER					HAS
361	APPROVED ENGINEERING ORDER					
	<>--- 333 ENGINEERING ORDER					BECOMES
	<>--- 14 ENG CHANGE					IS
	<>--- 415 PROCESS PLAN					AUTHORIZES
	<>--- 69 ENG RELEASE					IS
	<>--- 358 ENG ORDER USE FOR ENG CHANGE PROPOSAL					HAS
	<>--- 359 ENG ORDER USE FOR ENG CHANGE REQUEST					HAS
367	SPARE/KIT					
	<>--- 6 PART					IS
	<>--- 114 CONTRACT ITEM					REQUIRES
	<>--- 532 SPARE/MISC PART					REQUIRES
	<>--- 541 SCHEDULE FURNISHED ITEM					IS
	<>--- 512 UNIT TIME ASSEM					IS
368	MATERIAL					
	<>--- 556 P/L ITEM MATL CALLOUT					IS SPECIFIED BY
	<>--- 89 MATL ISSUE					HAS
	<>--- 382 MATERIAL NEED					SATISFIES
	<>--- 543 PICK LIST ITEM					IS REQUIRED BY
373	FACILITIES PLAN					
	<>--- 79 RESOURCE PLAN					IS
374	QUALITY ASSURANCE PLAN					
	<>--- 79 RESOURCE PLAN					IS
378	CUTTING SPECIFICATION					
	<>--- 392 CUTTING SPEC CALLOUT					IS SPECIFIED BY
379	VISUAL AID					
	<>--- 395 VISUAL AID CALLOUT					IS SPECIFIED BY
NODE:	TITLE:	RELATED	ENTITY CLASS	NODE CROSS	REFERENCE	NUMBER:

286	T63,T120	An instance of COORDINATE SYSTEM globally defines zero, one or many ITEM VERSION LOCATION
279	T63,T114	An instance of COORDINATE SYSTEM locally defines direction of zero, one or many ENVIRONMENT
277	T63,T103	An instance of COORDINATE SYSTEM locally defines zero, one or many MODEL NODE LOCATION
281	T63,T72	An instance of COORDINATE SYSTEM locally defines zero, one or many GEOMETRY ELEMENT
118	T64,T44	An OPERATIONAL ITEM VERSION is a function type of ITEM VERSION FUNCTION
421	T64,T224	An instance of OPERATIONAL ITEM VERSION has zero, one or many STORAGE LOCATION
424	T64,T192	An instance of OPERATIONAL ITEM VERSION has zero, one or many PACKAGE OPERATIONAL ITEM
423	T64,T172	An instance of OPERATIONAL ITEM VERSION has zero, one or many OPERATIONAL ITEM VERSION LSA TASK
422	T64,T178	An instance of OPERATIONAL ITEM VERSION has zero, one or many MAINTENANCE HISTORY
425	T64,T169	An instance of OPERATIONAL ITEM VERSION is used as equipment for zero, one or many LSA TASK OSE
130	T64,T38	An instance of OPERATIONAL ITEM VERSION relates to zero, one or many MANUFACTURING OPERATIONAL ITEM VERSION
132	T64,T36	An instance of OPERATIONAL ITEM VERSION relates to zero, one or many ENGINEERING OPERATIONAL ITEM VERSION
164	T65,T11	An instance of SIMULATED ITEM VERSION is modeled by exactly one ITEM VERSION
169	T65,T11	An instance of SIMULATED ITEM VERSION models exactly one ITEM VERSION
78	T66,T11	An instance of PACKAGE ITEM contains exactly one ITEM VERSION
76	T66,T29	An instance of PACKAGE ITEM is included in exactly one PACKAGE
248	T67,T63	An instance of ITEM VERSION ENVIRONMENT LOCATION has location defined by exactly one COORDINATE SYSTEM
250	T67,T11	An instance of ITEM VERSION ENVIRONMENT LOCATION is for exactly one ITEM VERSION
290	T67,T114	An instance of ITEM VERSION ENVIRONMENT LOCATION is for exactly one ENVIRONMENT

419	T29,T168	A PACKAGE may be a REPROCUREMENT PACKAGE	74
67	T29,T43	An instance of PACKAGE includes zero, one or many PACKAGE DOCUMENTED DATA	
75	T29,T66	An instance of PACKAGE includes zero, one or many PACKAGE ITEM	
580	T29,T192	An instance of PACKAGE is applicable to zero, one or many PACKAGE OPERATIONAL ITEM	
135	T30,T11	An instance of SERIALIZED ITEM VERSION is for exactly one ITEM VERSION	
111	T31,T39	An OPERATIONAL ITEM USAGE is a function type of ITEM USAGE FUNCTION	
61	T32,T51	An instance of INVOKED DOCUMENTED DATA REVISION is for exactly one DOCUMENTED DATA (DD) REVISION	
62	T32,T54	An instance of INVOKED DOCUMENTED DATA REVISION limits exactly one INVOKED DOCUMENTED DATA	
45	T33,T51	A MANUFACTURING AND OPERATIONAL ANALYSIS DOCUMENT is a type of DOCUMENTED DATA (DD) REVISION	
102	T34,T11	An instance of ITEM VERSION DOCUMENTED DATA REVISION defines, evaluates, and/or acts upon exactly one ITEM VERSION	
64	T34,T51	An instance of ITEM VERSION DOCUMENTED DATA REVISION is for exactly one DOCUMENTED DATA (DD) REVISION	
3	T35,T51	An ENGINEERING ORDER is a type of DOCUMENTED DATA (DD) REVISION	
127	T36,T16	An instance of ENGINEERING OPERATIONAL ITEM VERSION is related to exactly one ENGINEERING ITEM VERSION	
133	T36,T64	An instance of ENGINEERING OPERATIONAL ITEM VERSION is related to exactly one OPERATIONAL ITEM VERSION	
5	T37,T51	A PRODUCTION PLAN is a type of DOCUMENTED DATA (DD) REVISION	
129	T38,T24	An instance of MANUFACTURING OPERATIONAL ITEM VERSION is related to exactly one MANUFACTURING ITEM VERSION	
131	T38,T64	An instance of MANUFACTURING OPERATIONAL ITEM VERSION is related to exactly one OPERATIONAL ITEM VERSION	
113	T39,T25	An ITEM USAGE FUNCTION may be a MANUFACTURING ITEM USAGE	
112	T39,T3	An ITEM USAGE FUNCTION may be an ENGINEERING ITEM USAGE	

APPENDIX B

ISEPO Model

INTEGRATED INFORMATION SYSTEM EVOLUTION PROCESS (IIEEP) PROOF TREE

Dr. J. H. May 5, 1987

ACTIVITIES

A-1 OPERATE BUSINESS

A-1.1 DEVELOP BUSINESS STRATEGY

- A-1.1.1 EVALUATE STRATEGIC CONCEPTS
- A-1.1.2 DEFINE STRATEGIC OBJECTIVES
- A-1.1.3 PRIORITIZE STRATEGIC OBJECTIVES

A-1.2 ADMINISTER BUSINESS STRATEGY

- A-1.2.1 PLAN STRATEGIC PROJECTS
- A-1.2.2 AUTHORIZE STRATEGIC PROJECTS
- A-1.2.3 CONTROL STRATEGIC PROJECTS AGAINST TACTICAL PERFORMANCE

A-1.3 MONITOR EFFECTIVENESS OF BUSINESS STRATEGY

A-2 PLAN AND MANAGE STRATEGIC INTEGRATED INFORMATION SYSTEM PROJECTS

A-2.1 PLAN AND MANAGE INTEGRATED INFORMATION SYSTEM (IIS) EVOLUTION PROJECT

A-2.2 PLAN AND MANAGE IIS EVOLUTION PROJECT

- A-2.2.1 DEVELOP IIS EVOLUTION STRATEGY
- A-2.2.2 EFFECT IIS EVOLUTION
- A-2.2.3 OPERATE/MAINTAIN IIS

A-2.3 DEVELOP IIS EVOLUTION STRATEGY

A-2.3.1 INTERPRET IIS EVOLUTION PROJECT PROPOSAL

A-2.3.2 EVALUATE IMPACT OF IIS EVOLUTION PROJECT PROPOSAL ON BUSINESS STRATEGY

- A-2.3.2.1 EXAMINE MISSION, POLICIES AND LONG RANGE PLAN
- A-2.3.2.2 EXAMINE NEW PRODUCT/SERVICE CHARACTERISTICS
- A-2.3.2.3 EXAMINE PRODUCTION REQUIREMENTS
- A-2.3.2.4 EXAMINE MARKETING REQUIREMENTS
- A-2.3.2.5 EXAMINE RESOURCE REQUIREMENTS
- A-2.3.2.6 EXAMINE DISTRIBUTION REQUIREMENTS

A-2.3.3 EVALUATE POTENTIAL IMPACT OF IIS EVOLUTION ON BUSINESS OPERATIONS

- A-2.3.3.1 EXAMINE ORGANIZATION STRUCTURE
- A-2.3.3.2 EXAMINE FUNCTIONAL ARCHITECTURE

MECHANISMS

INFORMATION INTEGRATION RESOURCE MANAGEMENT/CONTROL SYSTEM

STRATEGIC PLANNING TOOL

- CONCEPT SPECIFICATION TOOL
- OBJECTIVE SPECIFICATION TOOL
- PRIORITIZATION TOOL

PROJECT MANAGEMENT AND CONTROL TOOL

- STRATEGIC PROJECT PLANNING TOOL
- " " " "
- STRATEGIC PROJECT CONTROL TOOL

EXECUTIVE MANAGEMENT COMMITTEE

INTEGRATED INFORMATION SYSTEM EVOLUTION TOOL

SAME AS THAT SPECIFIED FOR A-2

SAME AS THAT SPECIFIED FOR A-2

PROJECT DATA MANAGEMENT AND CONTROL TOOL
" " " "
IIS EXECUTIVE CONTROL SYSTEM &
DATA ACQUISITION SYSTEM

EXECUTIVE MANAGEMENT COMMITTEE, ENTERPRISE ANALYSIS TOOL, PROJECT DATA MANAGEMENT AND CONTROL TOOL

EXECUTIVE MANAGEMENT COMMITTEE

EXECUTIVE MANAGEMENT COMMITTEE

SAME AS THAT SPECIFIED FOR A-2

" " " "

" " " "

" " " "

" " " "

ENTERPRISE ANALYSIS TOOL

DECOMPOSITION ANALYSIS TOOL
INPUT/PROCESS/OUTPUT ANALYSIS TOOL

A133 EXAMINE STAFF SKILLS
 A134 EXAMINE SYSTEM ARCHITECTURE
 A135 EXAMINE INFORMATION FLOW
 A136 EXAMINE DATA ARCHITECTURE

A14 DEVELOP TACTICAL PLAN FOR IIS EVOLUTION

A141 DEFINE DEVELOPMENT PLAN

A1411 ESTABLISH APPLICATION PLAN
 A1412 ESTABLISH DATA PLAN
 A1413 ESTABLISH SYSTEM PLAN
 A1414 ESTABLISH PROJECT PLAN

A142 DEFINE MANAGEMENT PLAN

A1421 ESTABLISH MANAGEMENT SYSTEM PLAN
 A1422 ESTABLISH MANAGEMENT SYSTEM MONITORING PLAN

A143 DEFINE SERVICE PLAN

A1431 ESTABLISH SERVICE MARKETING PLAN
 A1432 ESTABLISH SERVICE LEVEL PLAN
 A1433 ESTABLISH RECOVERY PLAN
 A1434 ESTABLISH SECURITY PLAN
 A1435 ESTABLISH AUDIT PLAN

A144 DEFINE RESOURCE PLAN

A1441 ESTABLISH CAPACITY PLAN
 A1442 ESTABLISH SKILLS PLAN
 A1443 ESTABLISH BUDGET PLAN

A15 MANAGE DEVELOPMENT OF TACTICAL PLANS

A151 REVIEW PLANS
 A152 BALANCE PLANS
 A153 OBTAIN PRELIMINARY APPROVAL
 A154 PUBLISH PLANS

A16 ADMINISTER FINAL APPROVAL OF TACTICAL PLAN FOR IIS EVOLUTION

A2 EFFECT IIS EVOLUTION

A21 ADMINISTER IIS EVOLUTION PROJECT

A211 REVIEW TACTICAL PLAN FOR IIS EVOLUTION
 A212 SECURE RESOURCES FOR IIS EVOLUTION PROJECT
 A213 ACTIVATE TACTICAL PLAN FOR IIS EVOLUTION
 A214 MANAGE EXECUTION OF IIS EVOLUTION PROJECT

A2141 CONTROL PROJECT

A21411 PROVIDE PROJECT ASSIGNMENT

SKILLS ANALYSIS TOOL
 HARDWARE/SOFTWARE ANALYSIS TOOL
 DATA FLOW ANALYSIS TOOL
 ENTITY RELATIONSHIP ANALYSIS TOOL
 PROJECT DATA MANAGEMENT & CONTROL TOOL

SAME AS THAT SPECIFIED FOR A14

APPLICATION PLANNING TOOL
 DATA PLANNING TOOL
 SYSTEM PLANNING TOOL
 PROJECT PLANNING TOOL

PROJECT DATA MANAGEMENT & CONTROL TOOL

SAME AS THAT SPECIFIED FOR A142
 " " " "

SERVICE PLANNING TOOL

SERVICE MARKETING PLANNING TOOL
 SERVICE LEVEL PLANNING TOOL
 RECOVERY PLANNING TOOL
 SECURITY PLANNING TOOL
 AUDIT PLANNING TOOL

RESOURCE PLANNING TOOL

CAPACITY PLANNING TOOL
 SKILLS PLANNING TOOL
 BUDGET PLANNING TOOL

PROJECT DATA MANAGEMENT & CONTROL TOOL

SAME AS THAT SPECIFIED FOR A15
 " " " "
 " " " "
 " " " "

SAME AS THAT SPECIFIED FOR A15

PROJECT DATA MANAGEMENT & CONTROL TOOL

PROJECT DATA MANAGEMENT & CONTROL TOOL

SAME AS THAT SPECIFIED FOR A21
 " " " "
 " " " "
 " " " "
 PROJECT DATA MANAGEMENT & CONTROL TOOL
 INFORMATION MANAGEMENT SERVICES STAFF

PROJECT DATA MANAGEMENT & CONTROL TOOL

SAME AS THAT SPECIFIED FOR A2141

A21412 PROVIDE PROJECT SCHEDULING	"	"	"	"	"
A21413 PROVIDE PROJECT MONITORING	"	"	"	"	"
A21414 PROVIDE PROJECT REQUIREMENTS CONTROL	"	"	"	"	"
A21415 PROVIDE PROJECT EVALUATION	"	"	"	"	"
A2142 CONTROL EVOLUTION	PROJECT DATA MANAGEMENT & CONTROL TOOL				
A21421 PROVIDE APPLICATION/SOFTWARE DEVELOPMENT AND UPGRADE	SAME AS THAT SPECIFIED FOR A2142				
A21422 PROVIDE APPLICATION/SOFTWARE PROCUREMENT AND UPGRADE	"	"	"	"	"
A21423 PROVIDE HARDWARE/FACILITY INSTALLATION AND UPGRADE	"	"	"	"	"
A21424 PROVIDE TUNING/SYSTEMS BALANCING	"	"	"	"	"
A2143 CONTROL RESOURCES	PROJECT DATA MANAGEMENT & CONTROL TOOL				
A21431 PROVIDE RESOURCE CHANGE CONTROL	SAME AS THAT SPECIFIED FOR A2143				
A21432 PROVIDE RESOURCE AND DATA INVENTORY CONTROL	"	"	"	"	"
A2144 CONTROL SERVICES	PROJECT DATA MANAGEMENT & CONTROL TOOL				
A21441 PROVIDE PRODUCTION AND DISTRIBUTION SCHEDULING	SAME AS THAT SPECIFIED FOR A2144				
A21442 PROVIDE RESOURCE AND DATA PERFORMANCE CONTROL	"	"	"	"	"
A21443 PROVIDE PROBLEM CONTROL	"	"	"	"	"
A21444 PROVIDE SERVICE EVALUATION	"	"	"	"	"
A2145 PROVIDE ADMINISTRATIVE SERVICES	PROJECT DATA MANAGEMENT & CONTROL TOOL				
A21451 PROVIDE FINANCIAL ADMINISTRATION	SAME AS THAT SPECIFIED FOR A2145				
A21452 PROVIDE EDUCATION AND TRAINING	"	"	"	"	"
A21453 PROVIDE STAFF PERFORMANCE EVALUATION	"	"	"	"	"
A2146 PROVIDE INFORMATION MANAGEMENT SERVICES (IMS)	INFORMATION MANAGEMENT SERVICES STAFF				
A21461 PROVIDE IMS FOR PRODUCTION	SAME AS THAT SPECIFIED FOR A2146				
A21462 PROVIDE IMS FOR DISTRIBUTION	"	"	"	"	"
A21463 PROVIDE IMS FOR CUSTOMER SERVICE	"	"	"	"	"
A21464 PROVIDE IMS FOR SERVICE MARKETING	"	"	"	"	"
A215 MONITOR EFFECTIVENESS OF TACTICAL PLAN FOR ITS EVOLUTION	PROJECT DATA MANAGEMENT & CONTROL TOOL				
A216 MAINTAIN CUSTOMER INTERFACE	INFORMATION MANAGEMENT SERVICES STAFF				
A22 EVOLVE IIS	INFORMATION SYSTEM ENGINEERING TOOL				
A221 DEFINE PROBLEM	ANALYSIS TOOL,				
A2211 ANALYZE NEEDS	TECHNOLOGY ASSESSMENT TOOL,				
A22111 ANALYZE EXISTING SYSTEM	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL				
	SAME AS THOSE SPECIFIED FOR A221				
	FUNCTION MODELING TOOL,				
	INFORMATION MODELING TOOL,				
	DYNAMICS MODELING TOOL,				
	PROCESS MODELING TOOL,				
	SYSTEM ARCHITECTURE MODELING TOOL				
	PERFORMANCE ANALYSIS TOOL				

A221111 DEVELOP AS-IS ANALYSIS PLAN

A2211111 ESTABLISH AS-IS ANALYSIS MANAGEMENT TEAM

A2211112 DEVELOP WORKING PROCEDURES

A2211113 OBTAIN RESOURCES

A2211114 DEVELOP FUNCTIONAL GROUP INTERVIEW SCHEDULE

A2211115 DEVELOP INTERVIEW GUIDELINES AND DISTRIBUTE TO FUNCTIONAL GROUP MANAGEMENT PERSONNEL

A2211116 ESTABLISH ANALYSIS TASK FORCE AND EDUCATE IN THE USE OF ANALYSIS METHODOLOGICAL TECHNIQUES

A221112 MANAGE AS-IS ANALYSIS

A2211121 MONITOR PROGRESS OF AS-IS ANALYSIS

A2211122 DIRECT TECHNICAL EFFORT OF TASK FORCE

A2211123 CONDUCT STATUS MEETINGS

A2211124 PRODUCE AND DISTRIBUTE STATUS REPORTS

A221113 CONDUCT AS-IS ANALYSIS KICK-OFF MEETING

A2211131 INTRODUCE AS-IS ANALYSIS MANAGEMENT TEAM

A2211132 INTRODUCE TASK FORCE

A2211133 INTRODUCE FUNCTIONAL GROUP MANAGEMENT PERSONNEL

A2211134 REVIEW TACTICAL PLAN FOR IIS EVOLUTION

A2211135 REVIEW/REFINE AS-IS ANALYSIS PLAN

A2211136 REVIEW ANALYSIS METHODOLOGICAL TECHNIQUES

A221114 EXAMINE OPERATIONAL PROCEDURES DOCUMENTS AND TECHNICAL DATA TYPES

A221115 CONDUCT INTERVIEWS WITH FUNCTIONAL GROUP MANAGEMENT PERSONNEL

A2211151 IDENTIFY ORGANIZATION STRUCTURE

A2211152 IDENTIFY FUNCTIONAL ARCHITECTURE AND STAFF SKILLS

A2211153 IDENTIFY MANUAL AND AUTOMATED PROCESSES

A2211154 IDENTIFY SYSTEM ARCHITECTURE AND INFORMATION FLOW

A2211155 IDENTIFY DATA ARCHITECTURE

A2211156 IDENTIFY CURRENT OPERATIONAL PROBLEMS AND DESIRED OPERATIONAL CAPABILITIES

A221116 PRODUCE MODELS

A2211161 DEVELOP NODE TREE

A2211162 DEVELOP FUNCTION MODELS

A2211163 DEVELOP INFORMATION MODELS

A2211164 DEVELOP DYNAMICS (SIMULATION) MODELS

A2211165 DEVELOP PROCESS (STATE TRANSITION) MODELS

A2211166 DEVELOP SYSTEM ARCHITECTURE MODEL

PROJECT MANAGEMENT TEAM,
AS-IS ANALYSIS MANAGEMENT TEAM

PROJECT MANAGEMENT TEAM
AS-IS ANALYSIS MANAGEMENT TEAM

PROJECT MANAGEMENT TEAM,
AS-IS ANALYSIS MANAGEMENT TEAM

AS-IS ANALYSIS MANAGEMENT TEAM
"

" " " "

AS-IS ANALYSIS MANAGEMENT TEAM

AS-IS ANALYSIS MANAGEMENT TEAM
"

" " " "

AS-IS ANALYSIS MANAGEMENT TEAM

SAME AS THAT SPECIFIED FOR A221112
" " " " " "
" " " " " "
" " " " " "

AS-IS ANALYSIS MANAGEMENT TEAM

SAME AS THAT SPECIFIED FOR A221113
" " " " " "
" " " " " "
" " " " " "
" " " " " "

AS-IS ANALYSIS TASK FORCE

AS-IS ANALYSIS TASK FORCE,
ANALYSIS TOOLS

DECOMPOSITION ANALYSIS TOOL
INPUT/PROCESS/OUTPUT ANALYSIS TOOL,
SKILLS ANALYSIS TOOL
AS-IS ANALYSIS TASK FORCE
HARDWARE/SOFTWARE ANALYSIS TOOL,
DATA FLOW ANALYSIS TOOL
ENTITY RELATIONSHIP ANALYSIS TOOL
AS-IS ANALYSIS TASK FORCE

AS-IS ANALYSIS TASK FORCE,
MODELING TOOLS

NODE TREE MODELING TOOL
FUNCTION MODELING TOOL
INFORMATION MODELING TOOL
DYNAMICS MODELING TOOL
PROCESS MODELING TOOL
SYSTEM ARCHITECTURE MODELING TOOL

A22113 VALIDATE SYMPTOMS & CONCERNS	PERFORMANCE ANALYSIS TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22113 BOUND PROBLEM DOMAIN	
A22114 ASSESS TECHNOLOGY	SET ANALYSIS TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
	SURVEY TOOL, CODING & CLASSIFICATION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22115 IDENTIFY NEEDS	SYSTEM DESCRIPTION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22115 EVALUATE NEEDS	SET ANALYSIS TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A2212 DEFINE REQUIREMENTS	SAME AS THOSE SPECIFIED FOR A221
A22121 ASSESS TECHNOLOGY OPTIONS	SURVEY TOOL, CODING & CLASSIFICATION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22122 PARTITION PROBLEM DOMAIN	MODELING TOOL, SYSTEM DESCRIPTION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22123 IDENTIFY REQUIREMENTS	MODELING TOOL, SIMULATION TOOL, SYSTEM DESCRIPTION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22124 PERFORM TRIAL & ERROR EVALUATION OF REQUIREMENTS	PROTOTYPE DEVELOPMENT TOOL, SIMULATION TOOLS, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
	PROTOTYPE DEVELOPMENT TOOL
A221241 DEVELOP PROTOTYPES	
A22125 EVALUATE REQUIREMENTS	SET ANALYSIS TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22125E SOLUTION	CONFIGURATION MANAGEMENT TOOL, ANALYSIS TOOL, DESIGN TOOL, TECHNOLOGY ASSESSMENT TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A221 ESTABLISH DESIGN STRUCTURE	ANALYSIS TOOL, DESIGN TOOL, TECHNOLOGY ASSESSMENT TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A2211 EXAMINE REQUIREMENTS	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A2212 DEVELOP SOLUTIONS	MODELING TOOL, CONTROL FLOW ANALYSIS TOOL,

A22213 ASSESS APPLICABILITY OF TECHNOLOGY

DATA FLOW ANALYSIS TOOL,
SYSTEM DESCRIPTION TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

SURVEY TOOL,
SIMULATION TOOL,
CODING & CLASSIFICATION TOOL,
PROTOTYPE DEVELOPMENT TOOL
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22214 PERFORM TRADE-OFF ANALYSES OF SOLUTION ALTERNATIVES

SIMULATION TOOL,
COST/BENEFIT ANALYSIS TOOL
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A222141 PROVIDE ENGINEERING MAKE OR BUY DECISION

A22215 ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS

COST/BENEFIT ANALYSIS TOOL

SYSTEM DESCRIPTION TOOL,
PROTOTYPE DEVELOPMENT TOOL,
STRUCTURED DESIGN TOOL,
SIMULATION TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22216 EVALUATE DESIGN STRUCTURE

SET ANALYSIS TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A2222 DEVELOP SPECIFICATIONS

CONFIGURATION MANAGEMENT TOOL,
ANALYSIS TOOL,
DESIGN TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22221 DEFINE INTERNAL INTERFACES

STRUCTURED DESIGN TOOL,
SYSTEM DESCRIPTION TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22222 IDENTIFY SUBSYSTEMS AND COMPONENTS

LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22223 ANALYZE SUBSYSTEMS

SYSTEM DESCRIPTION TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22224 PACKAGE COMPONENTS

SYSTEM DESCRIPTION TOOL,
SIMULATION TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22225 ESTABLISH COMPONENT SPECIFICATIONS

SYSTEM DESCRIPTION TOOL,
DATABASE DESIGN TOOL,
SPECIFICATION LANGUAGES,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A22226 EVALUATE COMPONENT SPECIFICATIONS

CONFIGURATION MANAGEMENT TOOL,
SET ANALYSIS TOOL,
LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL

A222 BUILD SYSTEM

TEST TOOL,
DESIGN TOOL,
CONSTRUCTION TOOL,

CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 EVALUATION/SELECTION TOOL
 DESIGN TOOL,
 CONSTRUCTION TOOL,
 CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 STRUCTURED DESIGN TOOL,
 DATABASE DESIGN TOOL,
 SPECIFICATION LANGUAGE,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 CONSTRUCTION TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 SAME AS THOSE SPECIFIED FOR A22323
 CONSTRUCTION TOOL,
 CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 VERIFICATION TOOL,
 CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 VERIFICATION TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 SAME AS THOSE SPECIFIED FOR A22331
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 DESIGN TOOL,
 CONSTRUCTION TOOL,
 CONFIGURATION MANAGEMENT TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 CONSTRUCTION TOOL,
 LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
 SAME AS THOSE SPECIFIED FOR A22341
 STRUCTURED DESIGN TOOL,
 SYSTEM DESCRIPTION TOOL,
 CONFIGURATION MANAGEMENT TOOL,

A2231 PURCHASE COMPONENTS
 A2232 CONSTRUCT COMPONENTS

A22321 PREPARE DESIGN FOR IMPLEMENTATION

A22322 PLAN COMPONENT CONSTRUCTION
 A22323 FABRICATE PARTS
 A22324 ASSEMBLE PARTS
 A22325 ASSURE COMPONENT QUALITY

A2233 VERIFY SYSTEM ELEMENTS

A22331 PREPARE TEST ENVIRONMENT

A22332 EXERCISE VERIFICATION TEST CASES
 A22333 EVALUATE VERIFICATION MEASUREMENTS
 A22334 ESTABLISH VERIFICATION CONFORMANCE

A2234 INTEGRATE SYSTEM ELEMENTS

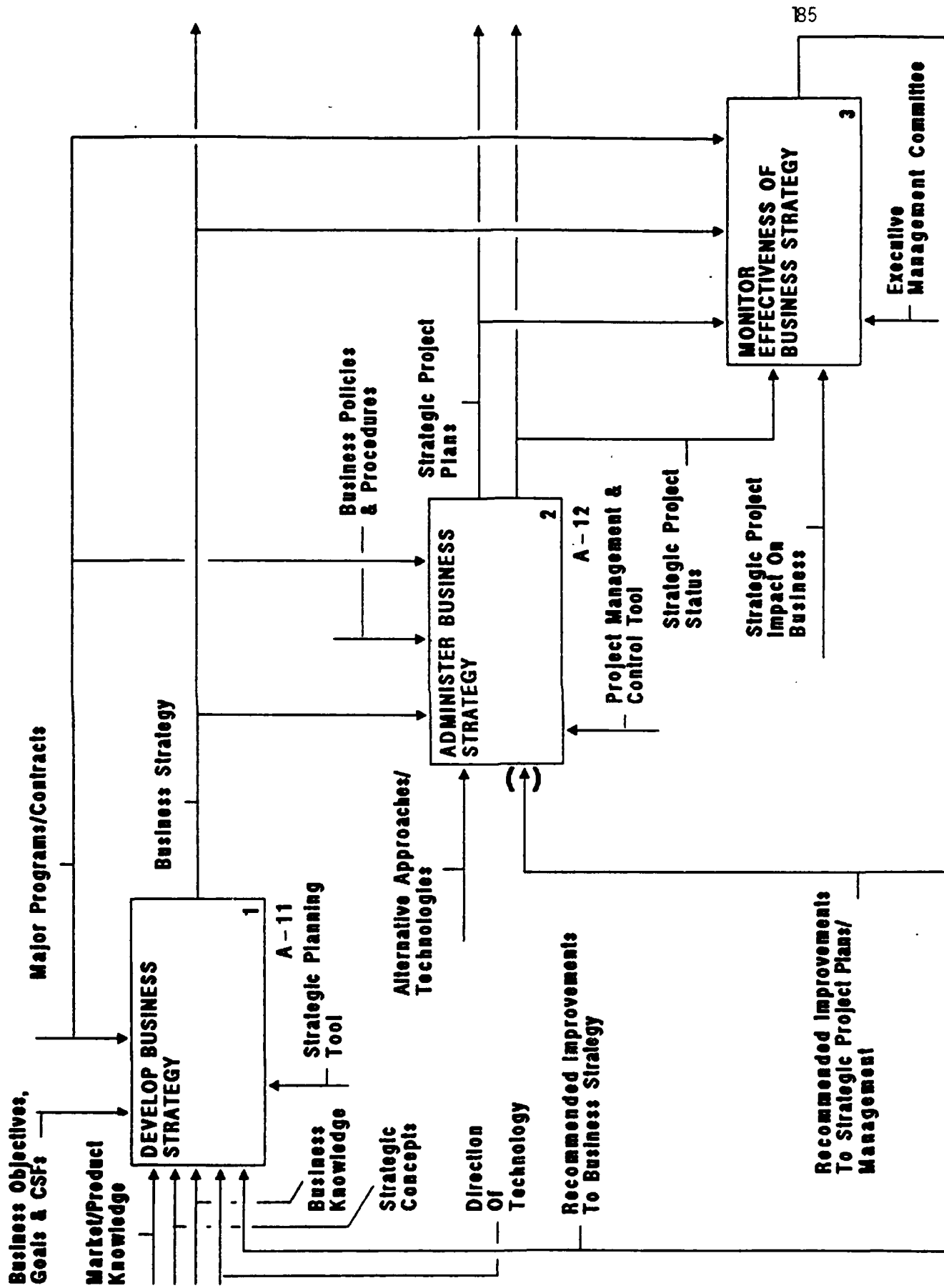
A22341 PLAN SYSTEM INTEGRATION

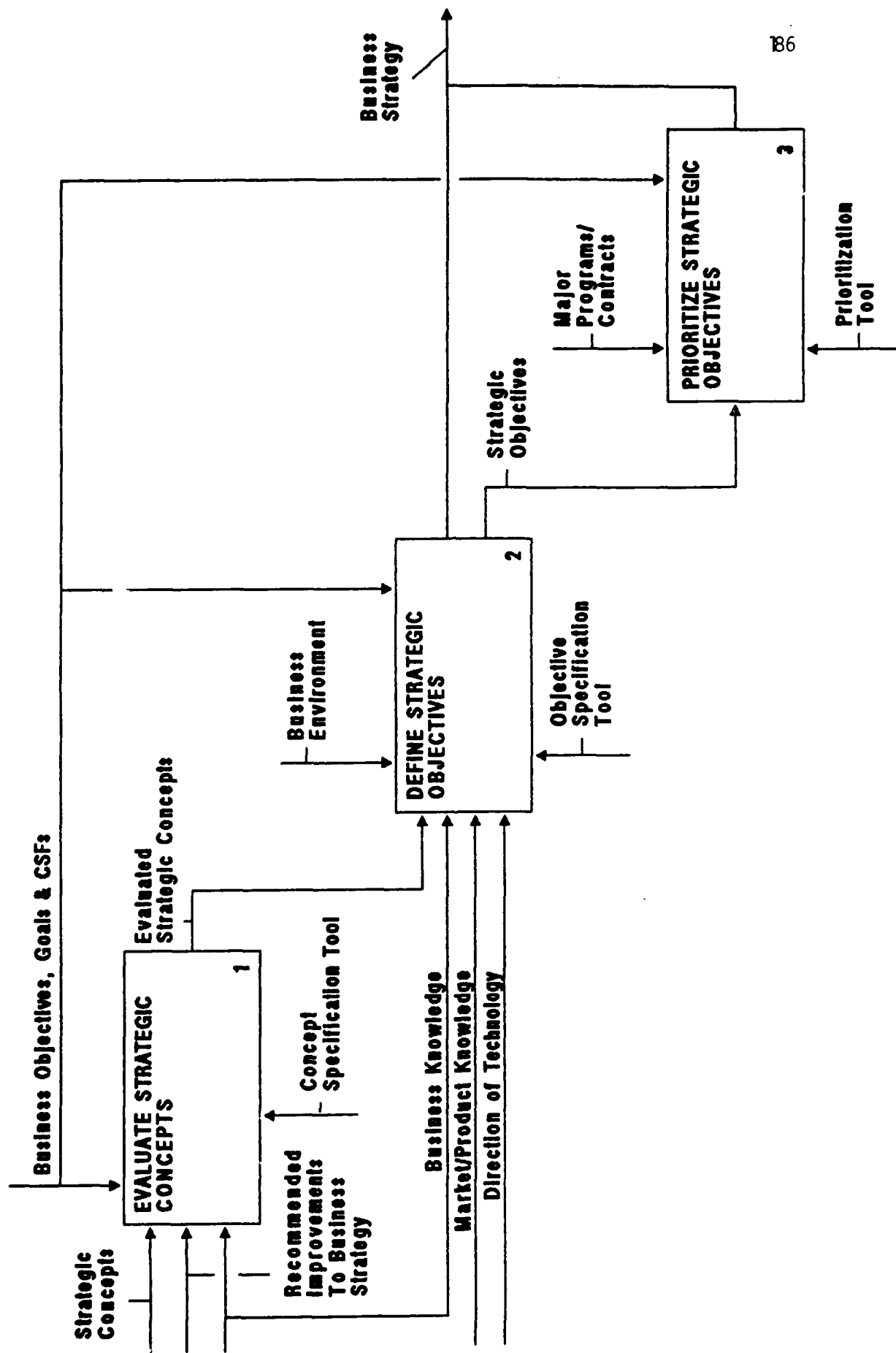
A22342 ASSEMBLE SYSTEM ELEMENTS

A22343 INTEGRATE OTHER SYSTEM ASPECTS
 A22344 RECORD AS-BUILT CONFIGURATIONS

A22345	PREPARE SYSTEM ELEMENTS FOR DELIVERY	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A2235	VALIDATE SYSTEM ELEMENTS	NONE SPECIFIED
A22351	PREPARE VALIDATION TEST ENVIRONMENT	VALIDATION TOOL, CONFIGURATION MANAGEMENT TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22352	EXERCISE VALIDATION TEST CASES	VALIDATION TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22353	EVALUATE VALIDATION MEASUREMENTS	SAME AS THOSE SPECIFIED FOR A22351
A22354	ESTABLISH VALIDATION CONFORMANCE	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A224	IMPLEMENT SYSTEM	CONFIGURATION MANAGEMENT TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A2241	ESTABLISH SYSTEM	CONFIGURATION MANAGEMENT TOOL, ANALYSIS TOOL, TEST TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22411	PREPARE ENVIRONMENT	CONFIGURATION MANAGEMENT TOOL, TEST TOOL, LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22412	CUSTOMIZE KNOWLEDGE BASE	NONE SPECIFIED
A224121	DEFINE PROJECTS	INFORMATION INTEGRATION SYSTEM
A224122	DEFINE ORGANIZATIONS	SAME AS THAT SPECIFIED FOR A22412
A224123	DEFINE ORGANIZATION STRUCTURES	" " " " " " " "
A224124	DEFINE PROJECTS	" " " " " " " "
A224125	DEFINE OBJECT STRUCTURES	" " " " " " " "
A224126	DEFINE OBJECT DATA FIELDS	" " " " " " " "
A224127	DEFINE LEGAL VALUE SETS	" " " " " " " "
A224128	DEFINE USER FORMS	" " " " " " " "
A224129	DEFINE OBJECT LIFE CYCLE STATES	" " " " " " " "
A2241210	DEFINE CONSTRAINT RULES	" " " " " " " "
A2241211	DEFINE ACCESS REASON RULES	" " " " " " " "
A2241212	DEFINE REVISION REASON RULES	" " " " " " " "
A2241213	DEFINE NOTIFICATION RECIPIENTS	" " " " " " " "
A2241214	DEFINE APPROVERS	" " " " " " " "
A2241215	DEFINE TASK INITIATION RULES	" " " " " " " "
A2241216	DEFINE PROCESS CONTROL RULES	" " " " " " " "
A2241217	DEFINE MENU ACCESS RULES	" " " " " " " "
A2241218	DEFINE FILE LOCATION RULES	" " " " " " " "
A22412	INSTALL SYSTEM	NONE SPECIFIED
A22413	TRAIN USERS	NONE SPECIFIED
A22414	INITIATE MAINTENANCE SUBSYSTEM	NONE SPECIFIED

A22-015 EVALUATE AS-INSTALLED SYSTEM	CONFIGURATION MANAGEMENT TOOL ,
	TEST TOOL ,
	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A22-016 OBTAIN USER ACCEPTANCE	NONE SPECIFIED
	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A23 PROVIDE SYSTEMS ENGINEERING ENVIRONMENT	SOFTWARE ENGINEERING TOOL
A24 PROVIDE TOOLING, FIRMWARE & DOCUMENTATION	SOFTWARE ENGINEERING TOOL
A3 OPERATE/MAINTAIN IIS	INSTALLED SYSTEM
	SYSTEM USER,
	OPERATION & MAINTENANCE TOOLS
A31 OPERATE SYSTEM	INSTALLED SYSTEM,
	SYSTEM USERS,
	OPERATION TOOLS
A32 MAINTAIN SYSTEM	CONFIGURATION MANAGEMENT TOOL ,
	ANALYSIS TOOL,
	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A321 MAKE SYSTEM MODIFICATIONS	PERFORMANCE ANALYSIS TOOL
A322 RECORD OPERATIONAL SYSTEM CONFIGURATION	CONFIGURATION MANAGEMENT TOOL
A323 IDENTIFY SYMPTOMS & CONCERNS	PERFORMANCE ANALYSIS TOOL
	LIFE-CYCLE ARTIFACT DATA MANAGEMENT TOOL
A324 ANALYZE PROBLEMS	SET ANALYSIS TOOL,
A325 UPDATE KNOWLEDGE BASE	INFORMATION INTEGRATION SYSTEM





B6

A-11 DEVELOP BUSINESS STRATEGY

AD-A195 851

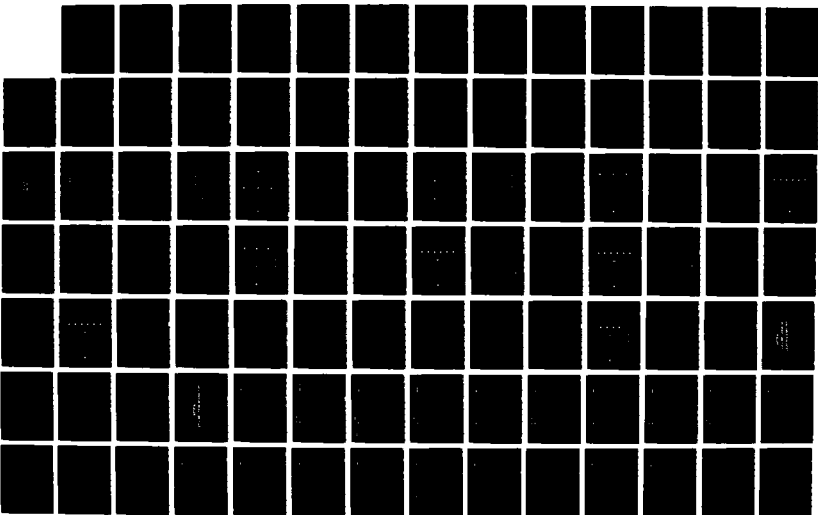
KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS
DEVELOPMENT METHODOLOGIES.. (U) MASSACHUSETTS INST OF
TECH CAMBRIDGE A GUPTA ET AL. DEC 87 MIT-KBIIS-2
DTR557-85-C-00003

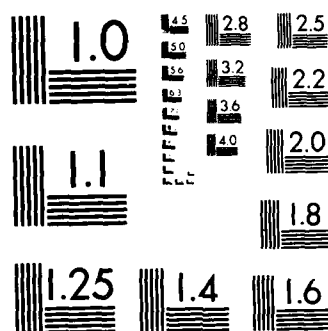
3/4

UNCLASSIFIED

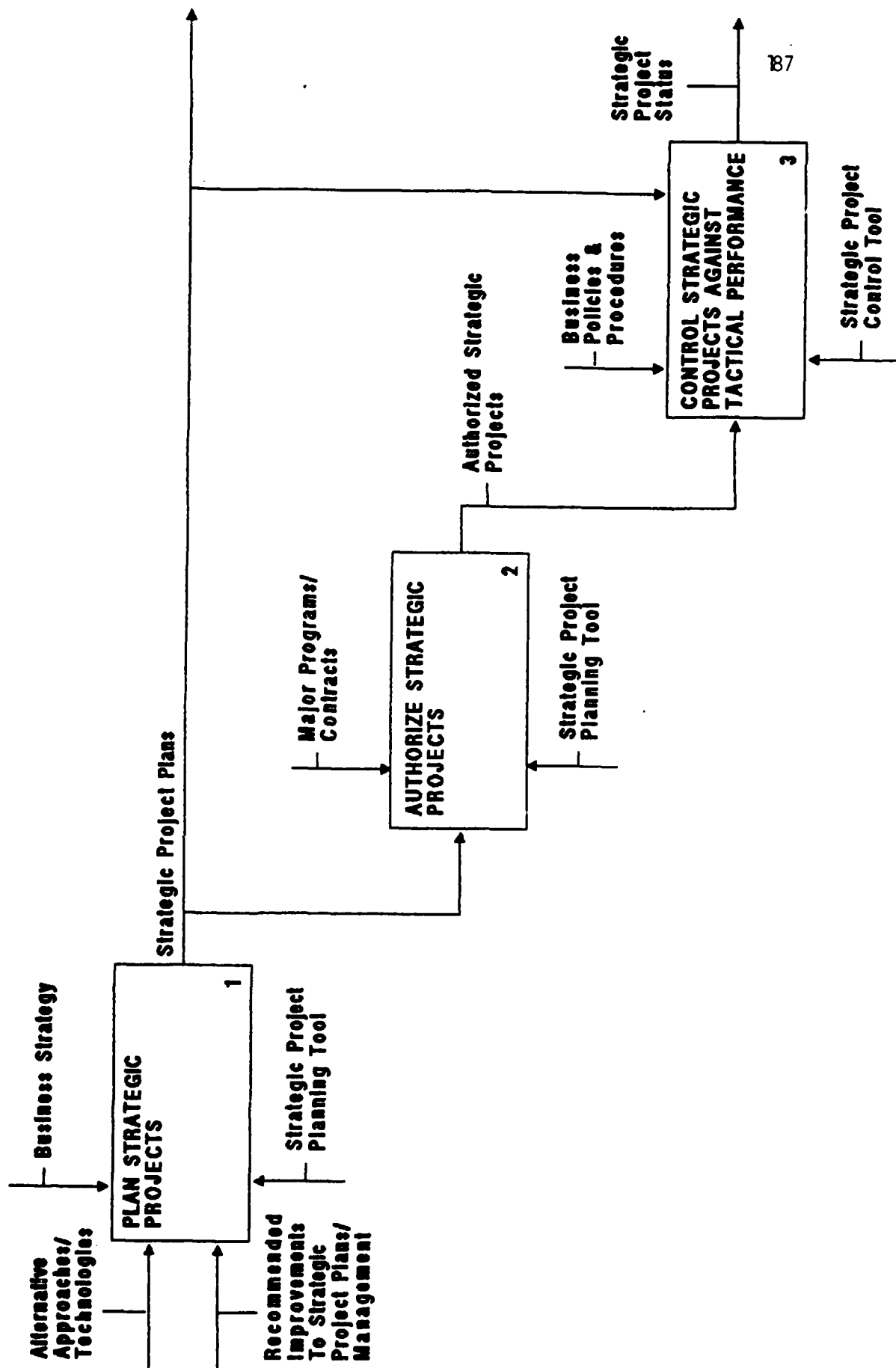
F/G 12/7

NL

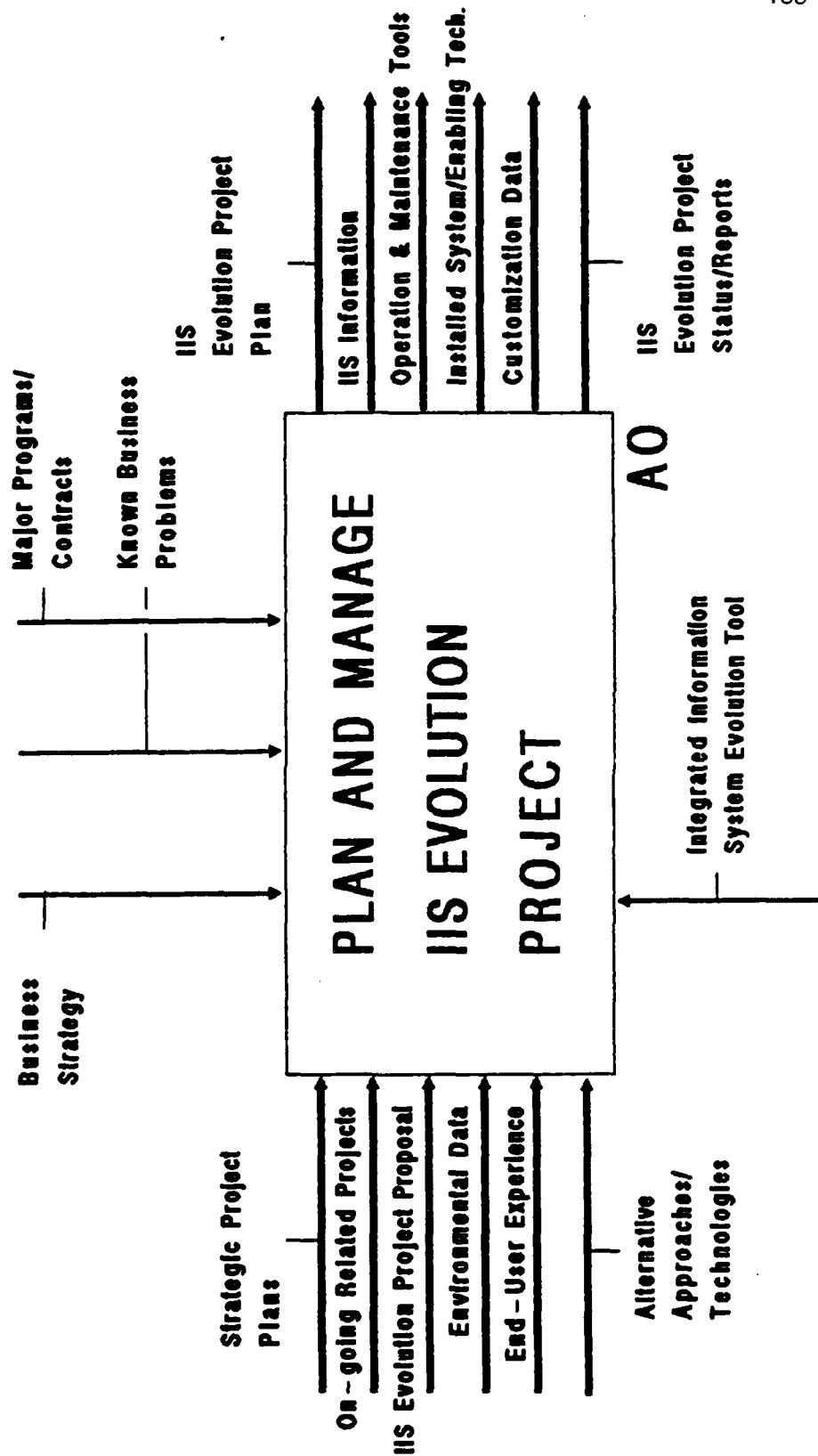


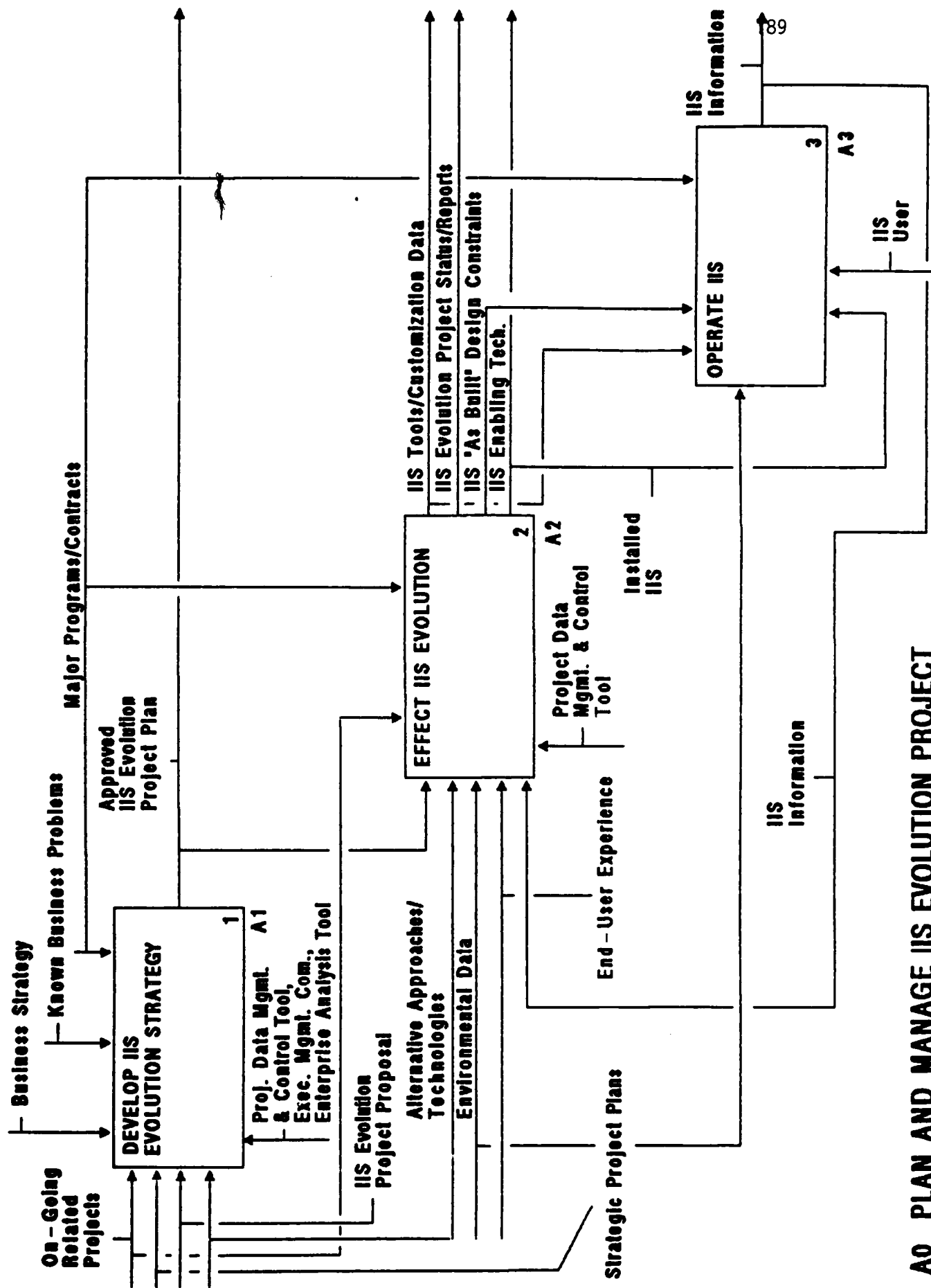


MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

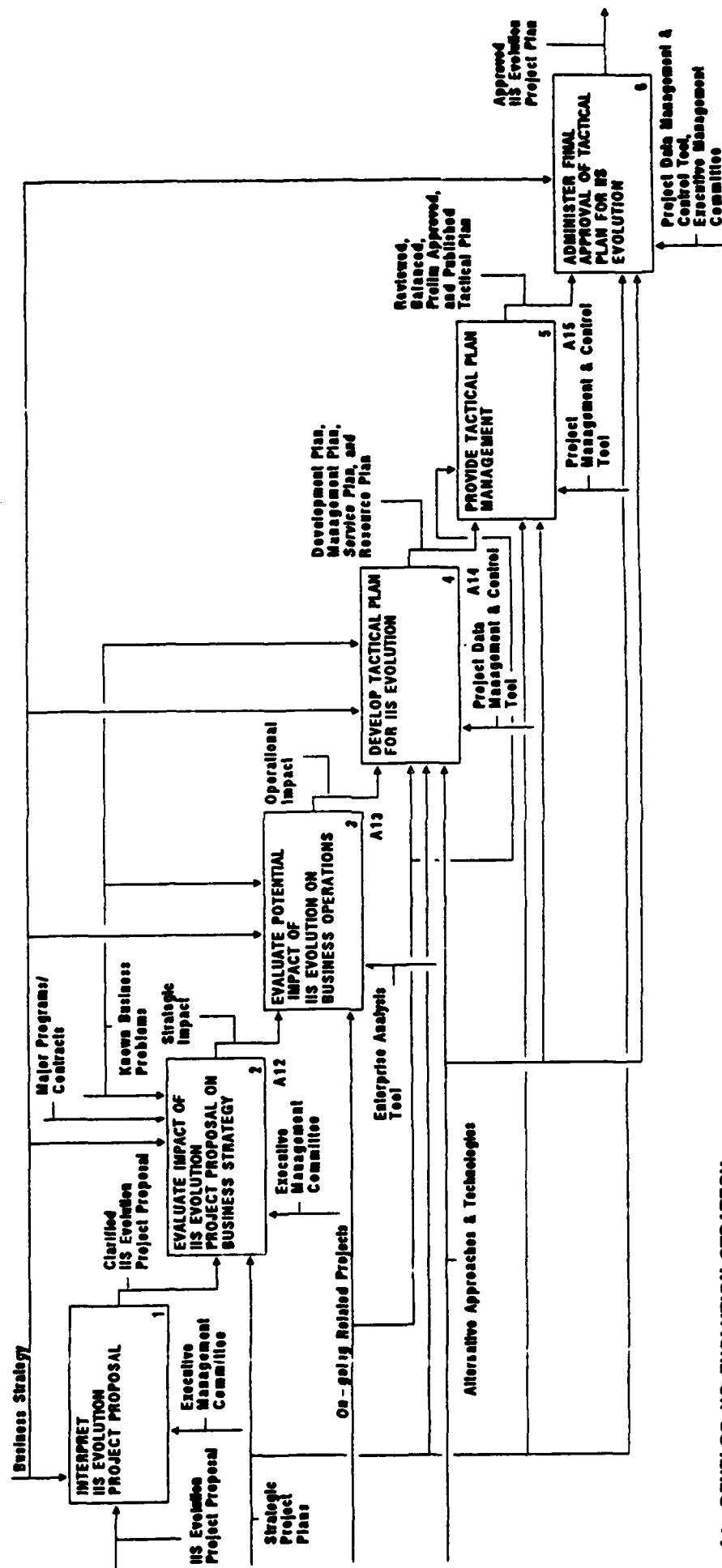


A-12 ADMINISTER BUSINESS STRATEGY

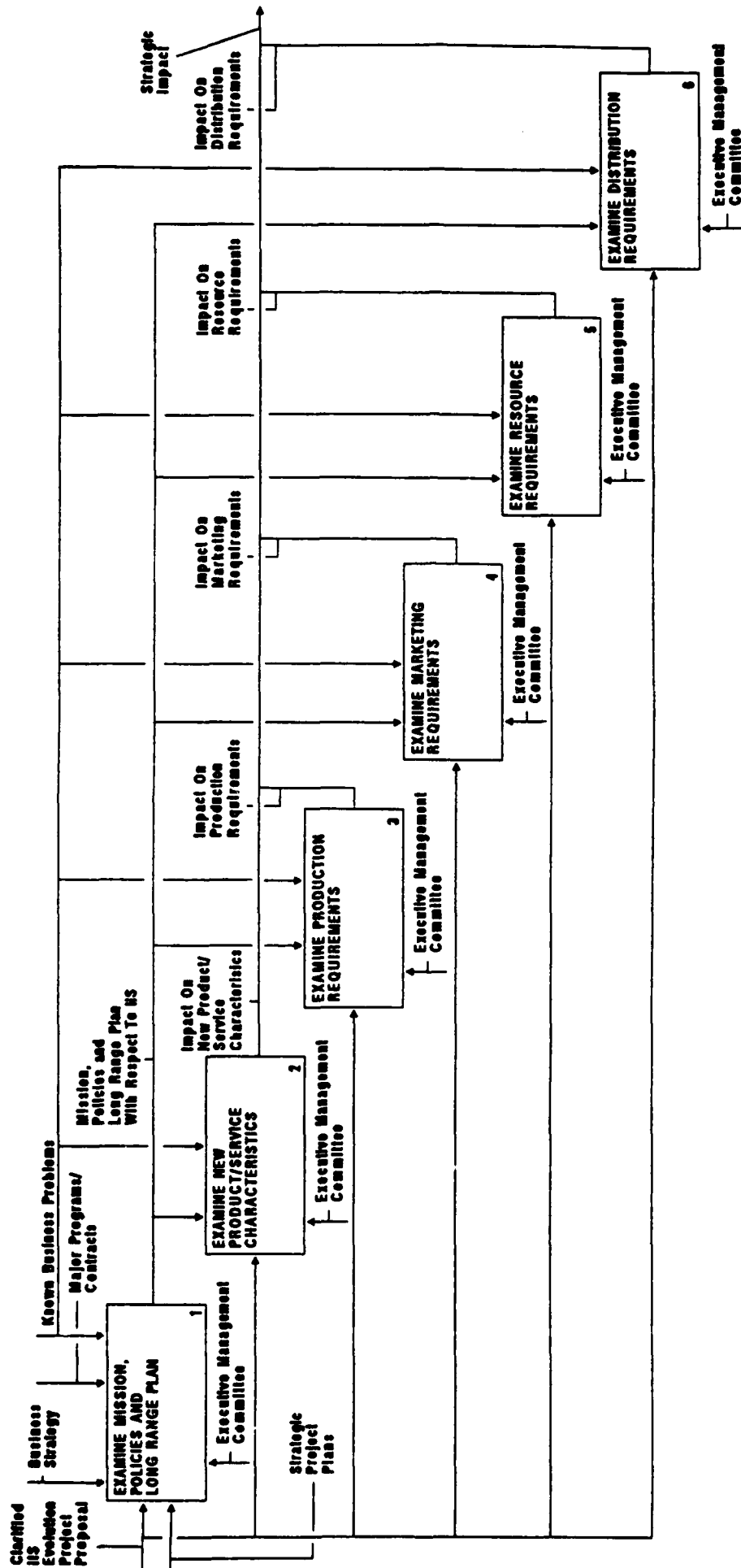




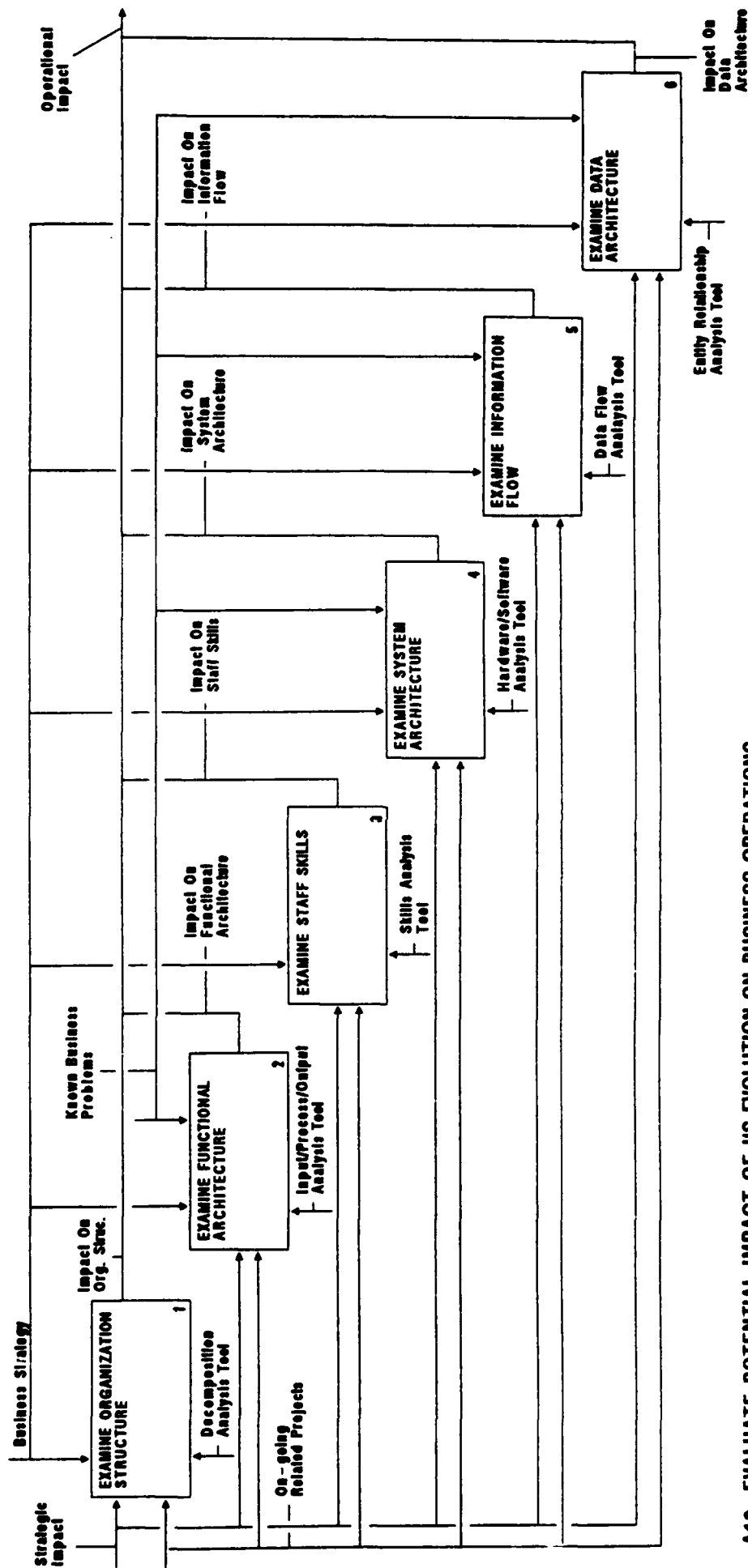
A0 PLAN AND MANAGE IIS EVOLUTION PROJECT



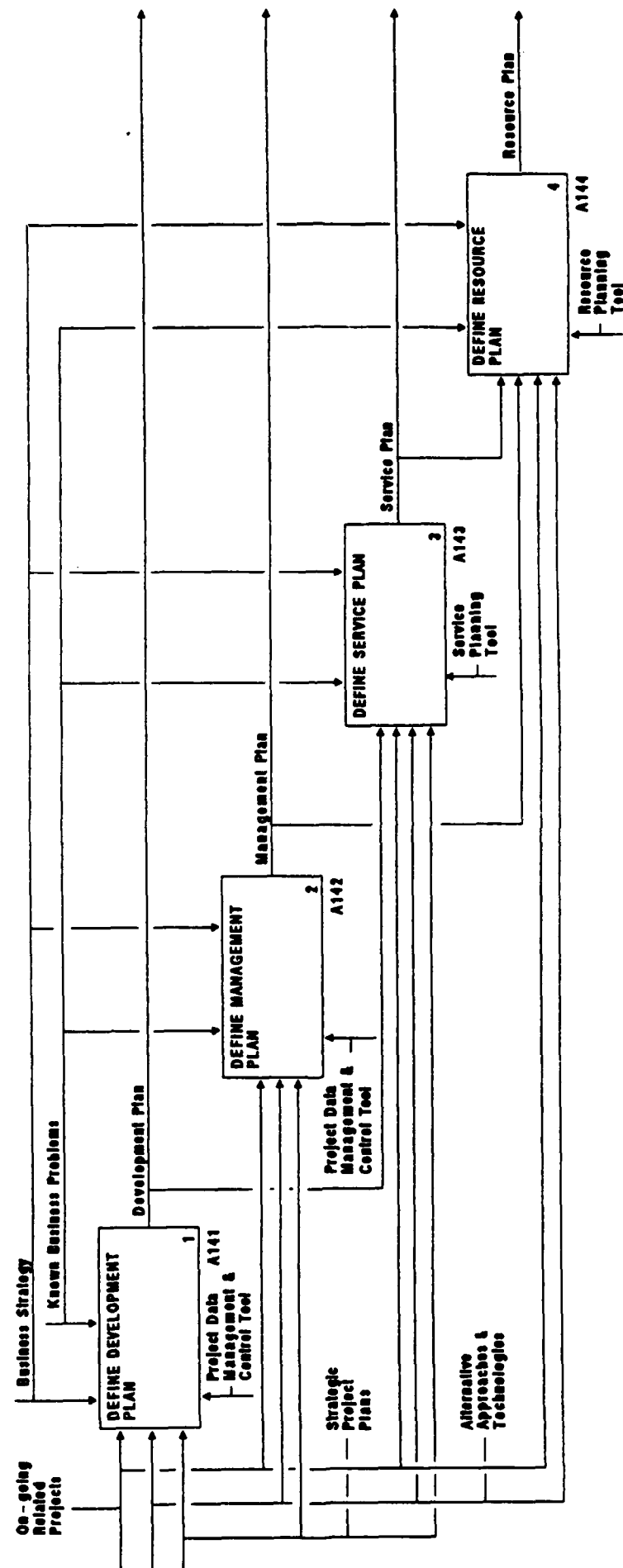
A1 DEVELOP IIS EVOLUTION STRATEGY



A12 EVALUATE IMPACT OF IIS EVOLUTION PROJECT PROPOSAL ON BUSINESS STRATEGY

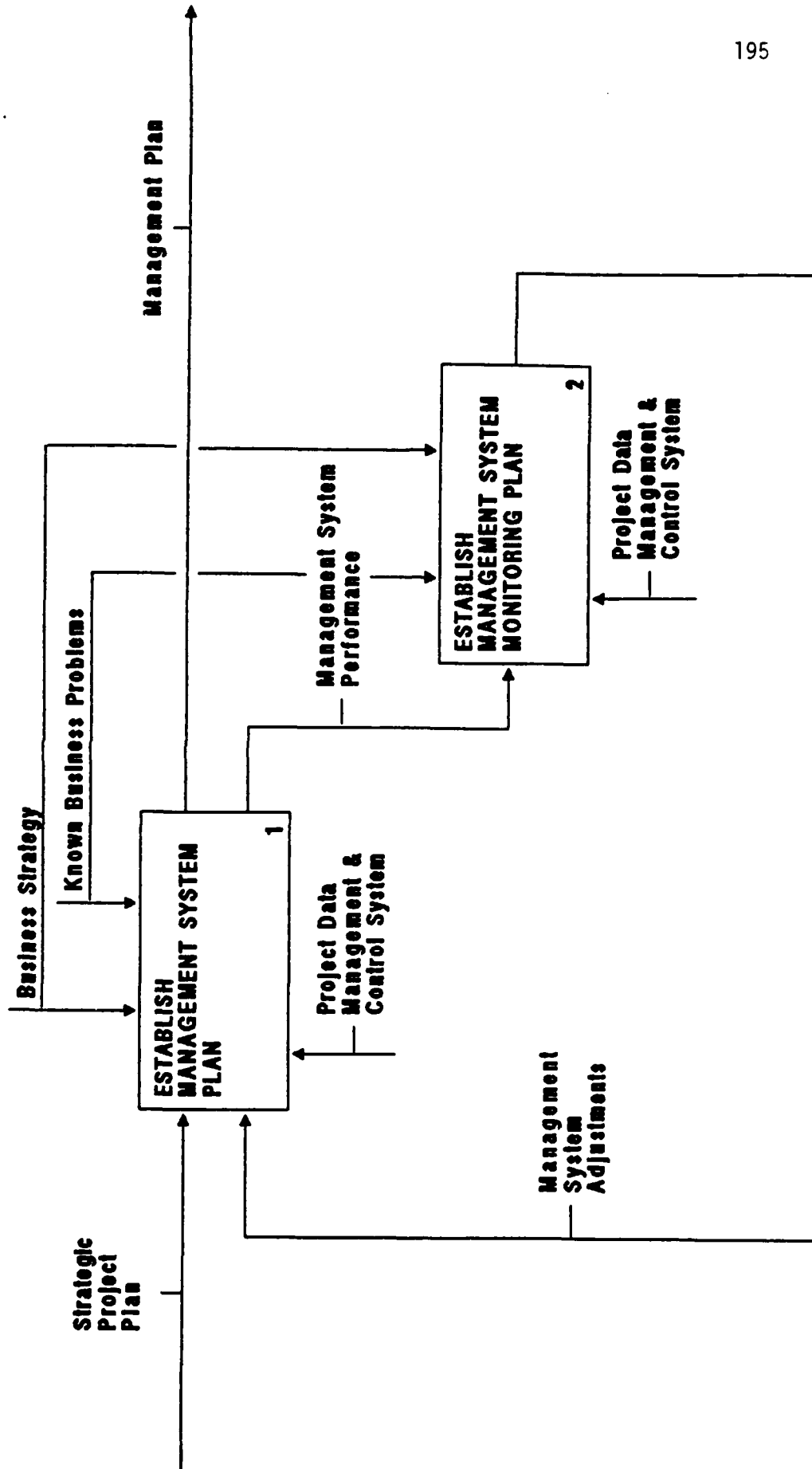


A13 EVALUATE POTENTIAL IMPACT OF IIS EVOLUTION ON BUSINESS OPERATIONS

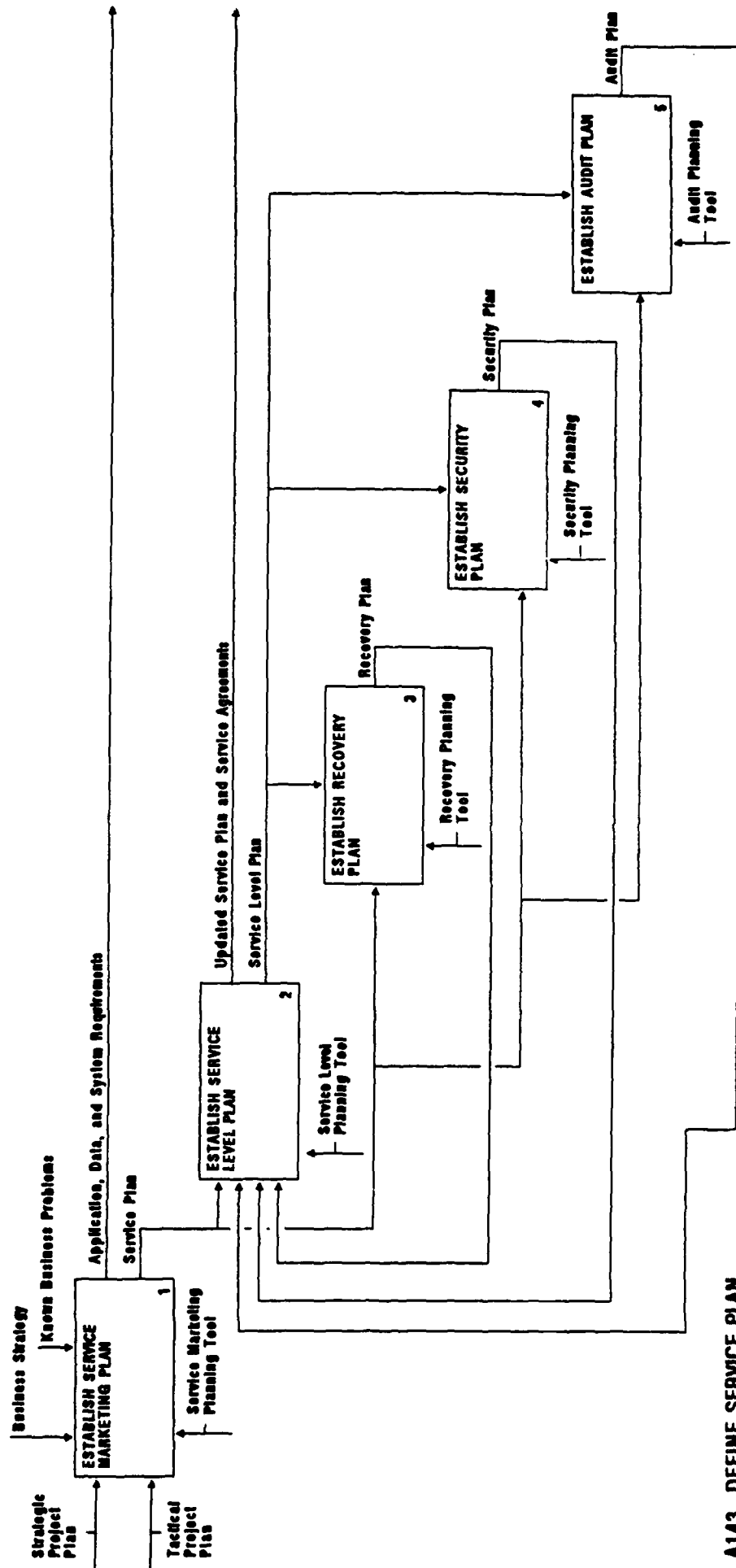


A14 DEVELOP TACTICAL PLAN FOR IIS EVOLUTION

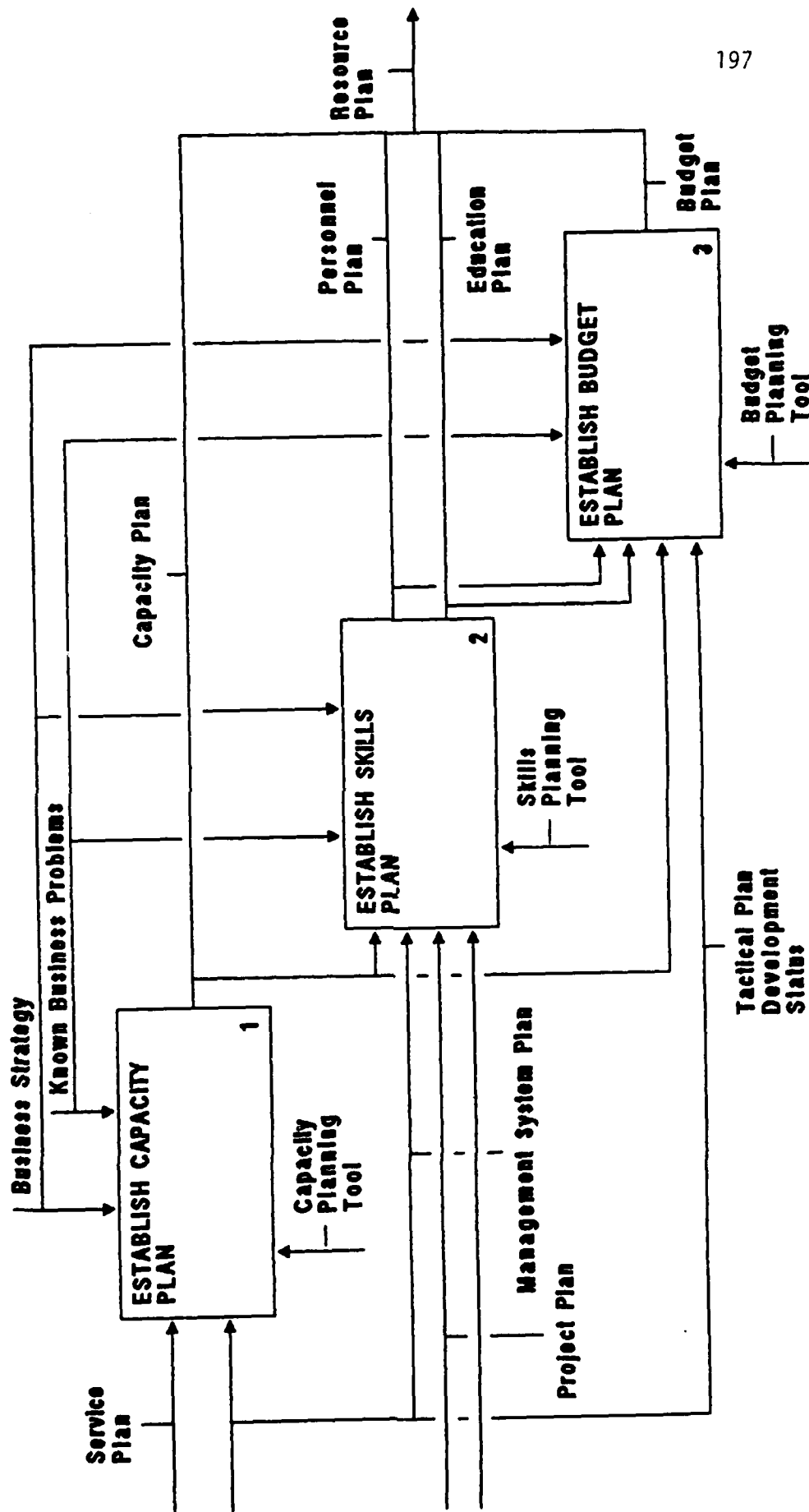




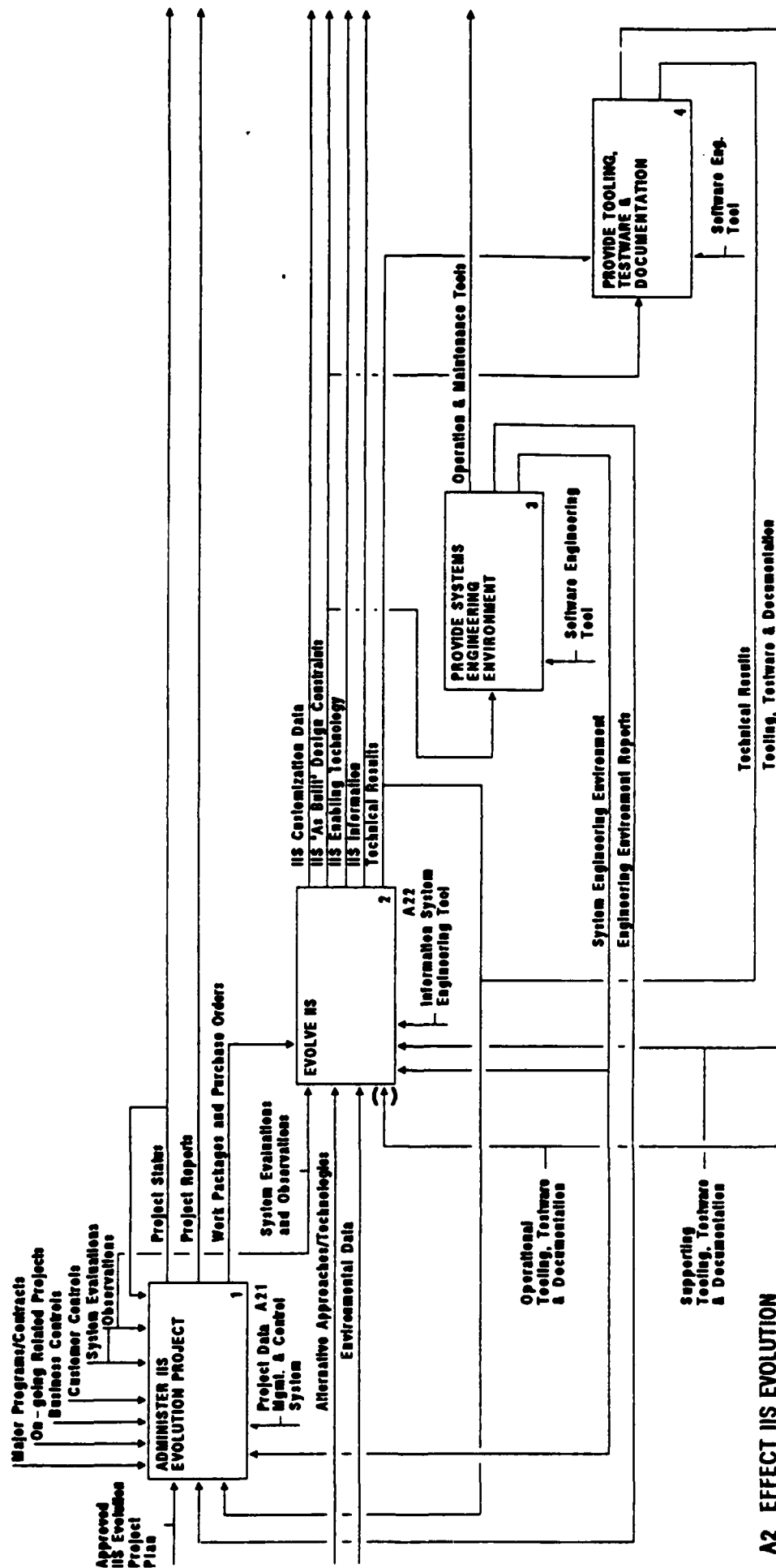
A142 DEFINE MANAGEMENT PLAN



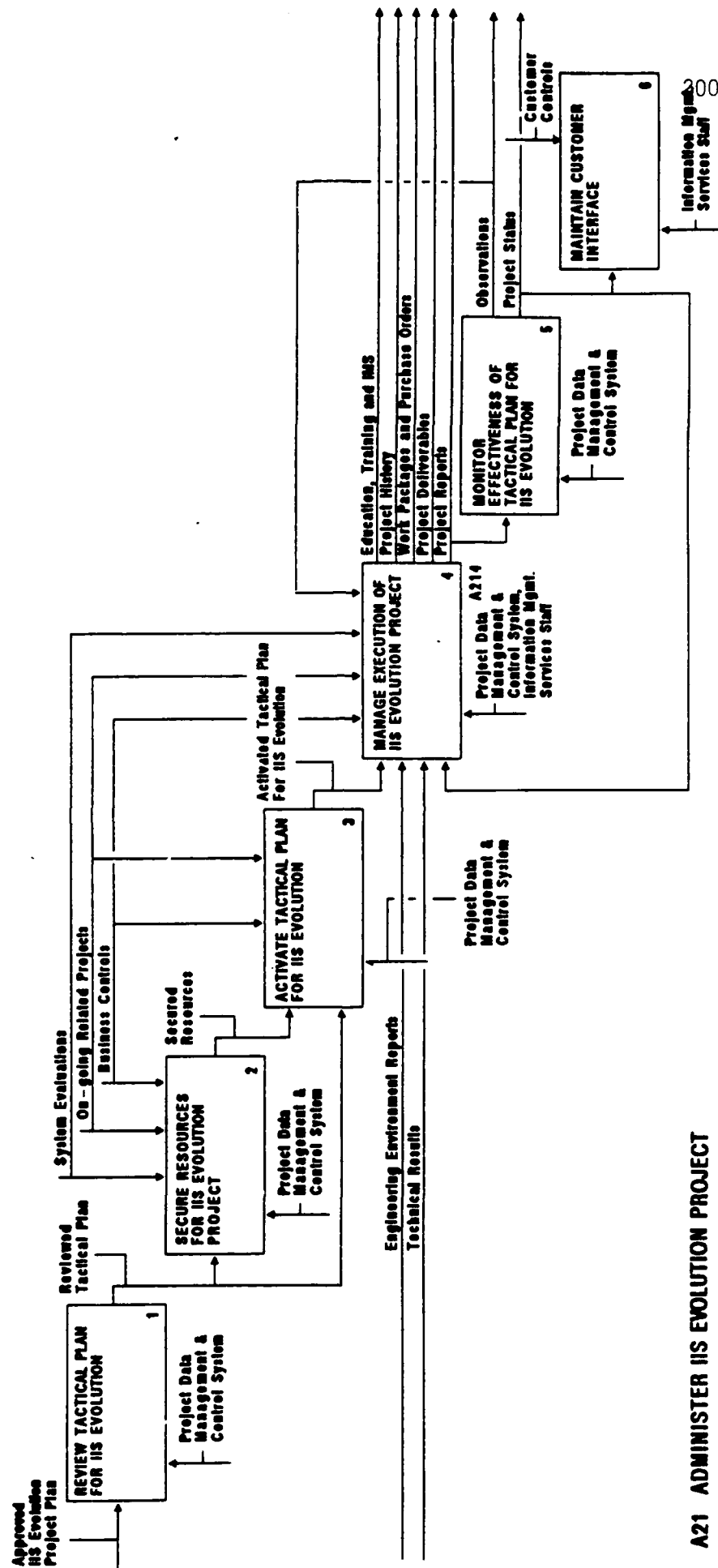
A143 DEFINE SERVICE PLAN



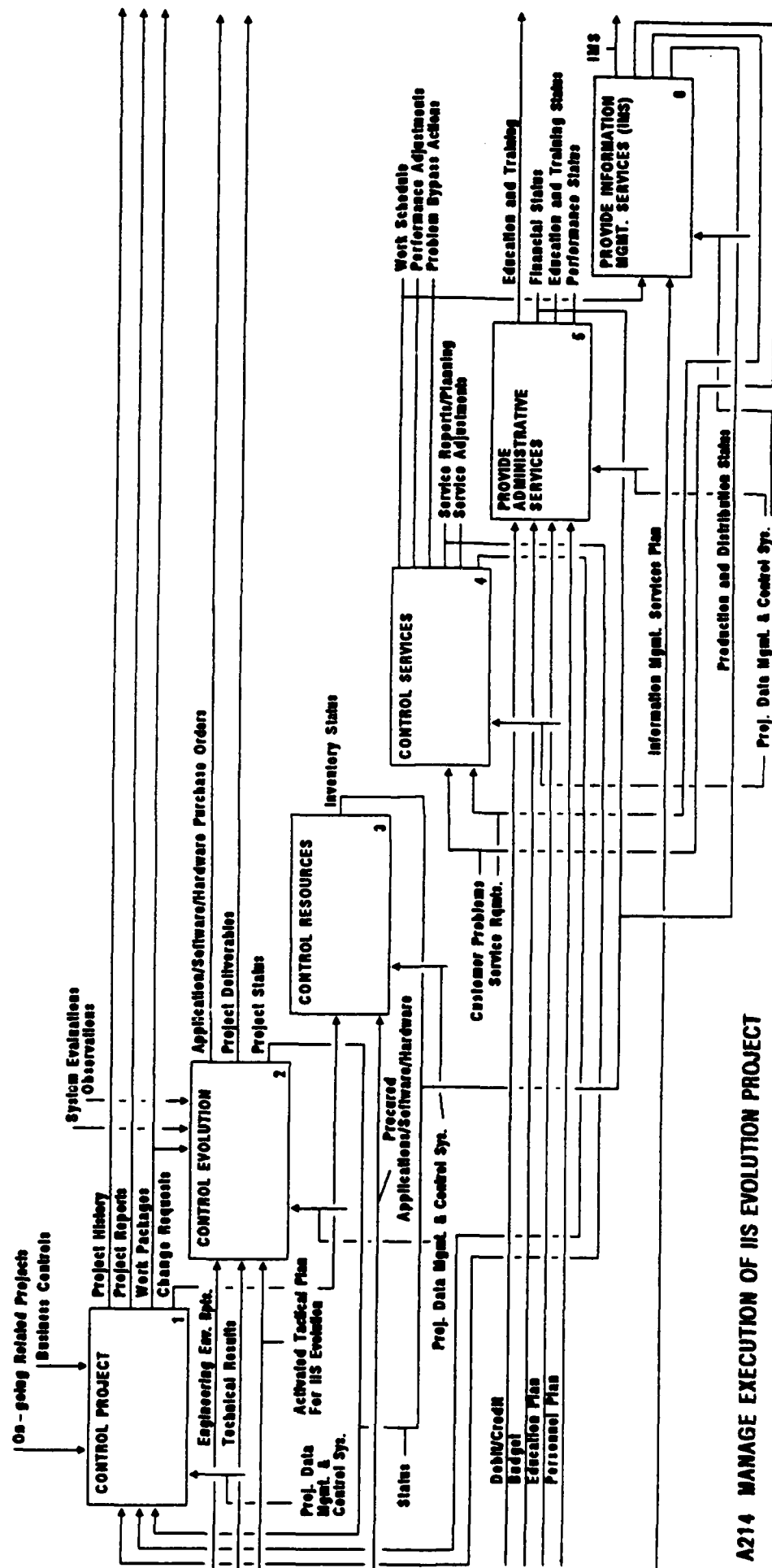




A2 EFFECT IIS EVOLUTION



A21 ADMINISTER IIS EVOLUTION PROJECT



A214 MANAGE EXECUTION OF IIS EVOLUTION PROJECT

(DRAFT)

A-1 OPERATE BUSINESS

This activity covers the development, administration and monitoring of the business strategy. The business strategy includes the goals that the enterprise intends to attain and the direction that the enterprise must take in order to achieve these goals.

A-11 DEVELOP BUSINESS STRATEGY

This activity focuses on the development of a workable business strategy. The first step is to evaluate strategic concepts. The concept evaluation process lays the groundwork for a more comprehensive definition of strategic goals of the enterprise. Strategic concepts are evaluated and chosen to be incorporated into the business strategy as strategic goals. Once the strategic goals have been thoroughly defined and final approval has been given as to their validity, they are prioritized for initiation.

A-12 ADMINISTER BUSINESS STRATEGY

Prioritized strategic goals of the enterprise are utilized during this activity to plan strategic projects. It is through these projects that the business intends to attain its strategic goals. Planning strategic projects is carried-out at a high level and includes the definition of such items as: deliverables, schedules, resources, cost, etc. Once the planning exercise has been completed, a subset of the developed strategic project plans are authorized for initiation. If selected for initiation, a strategic project plan is further developed and refined into one or more tactical project plans. The tactical project plans include all the specific information that is necessary to start-up and carry-out a project. The underlying purpose is to detail the methods for securing the objectives of the strategy. Finally, authorized strategic projects will be controlled based upon the performance of the tactical projects.

A-13 MONITOR EFFECTIVENESS OF BUSINESS STRATEGY

During this activity, the business strategy is monitored in two ways, i.e., in a bottom-up as well as a top-down fashion. As for bottom-up, on-going tactical projects are tracked in terms of their performance. They should operate effectively and attain the goals that are outlined in the strategic project plans. With top-down, changing technology and industry needs must be carefully analyzed so that strategic objectives can be modified accordingly.

A-0 PLAN AND MANAGE STRATEGIC INTEGRATED INFORMATION SYSTEM PROJECTS

Various strategic projects may be initiated by an enterprise in order to achieve strategic objectives. This particular activity focuses on the strategic planning/management that is associated with integrated information system projects.

A0 PLAN AND MANAGE INTEGRATED INFORMATION SYSTEM (IIS) EVOLUTION PROJECT

This activity focuses on the planning/management of the Integrated Information System (IIS) evolution project. The objective of the IIS project is to design, develop, construct, and demonstrate a prototype that effectively captures, manages, and distributes key digital and graphic technical data across the entire life span of a commercial product (e.g., automobile, airplane, etc.).

A1 DEVELOP IIS EVOLUTION STRATEGY

This activity focuses on the development of a strategy for the evolution of IIS. This cannot be accomplished without a thorough understanding of the IIS project proposal, its impact on the business strategy and the impact of the project on business operations. Once this has been determined, a tactical plan is developed, the development is managed and the final tactical plan is approved.

A12 EVALUATE IMPACT OF IIS EVOLUTION PROJECT PROPOSAL ON BUSINESS STRATEGY

Once the IIS project proposal has been interpreted, its impact on the overall business strategy must be evaluated. This would include an examination of the following: possible conflicts with the enterprise mission, policies and long range plan; new product/service characteristics and how they may affect, production, marketing, resources, etc.

A13 EVALUATE IMPACT OF IIS EVOLUTION ON BUSINESS OPERATIONS

Factors associated with the strategic impact that the IIS proposal has on the overall business strategy will filter down to impact business operations to some extent. The degree of impact must be evaluated through an examination of apparent changes to the existing organization structure, functional architecture, staff skills, system architecture, information flow and data architecture.

A14 DEVELOP TACTICAL PLAN FOR IIS EVOLUTION

This activity focuses on balancing the demand for resources for the IIS project with those for on-going projects.

This activity focuses on the the definition of the development plan for the IIS project. The development plan is made up of an application plan, data plan, service plan, and resource plan. The purpose of the development plan is to define the IIS project as an implementable subset of the strategic plan. The plan is defined such that it is carried-out over the tactical time period.

A1411 ESTABLISH APPLICATION PLAN

Based upon key functions of the enterprise and existing applications, this activity translates the IIS evolution strategy into an application plan. The application plan is a guide to be used for all application development and modification of system requirements that are to be met during the tactical time period. The plan focuses on a consistent application architecture so that the needs of all enterprise organizations can be satisfied.

A1412 ESTABLISH DATA PLAN

Based upon the application plan and present status of data, this activity translates the IIS evolution strategy into a data plan. The data plan focuses on data and data resources that exist within the enterprise. It defines requirements that are to be met during the tactical time period. The plan modifies and/or expands the existing data architecture in relation to the changes that are made to the application architecture because of the IIS evolution project.

A1413 ESTABLISH SYSTEMS PLAN

Based upon the application and data plans, the present status of hardware, software, the network, and the request for existing service modifications, this activity translates the IIS evolution strategy into a systems plan. The systems plan is develop utilizing the results of an examination of various technology alternatives. It defines requirements (i.e., new systems and/or integration of existing systems) that are to be met during the tactical time period. It identifies the functional environment for both application and data planning. The plan expands the systems architecture in relation to the changes that are made to the application and data architectures because of the IIS evolution project.

A1414 ESTABLISH PROJECT PLAN

This activity translates application, data, and systems plans, into well-defined and manageable IIS projects. A project plan is defined and applied to all new IIS related development. The plan contains a list of activities and tasks that are defined with respect to expected results. It includes pre-defined milestones which represent decision

points in the project. For each milestone task, resources and time is estimated so that reasonable decisions pertaining to results and associated risks can be made. Definitions of any unique project related control requirements are also included in the plan.

Once a reasonable project plan has been established, it must be financial justified. This is accomplished through an investment evaluation that estimates the costs and benefits resulting from the implementation and use of applications, data, and systems.

A142 DEFINE MANAGEMENT PLAN

This activity focuses on the creation, management and modification of the management system.

A1421 ESTABLISH MANAGEMENT SYSTEM PLAN

Using strategic guidance and an assessment of the existing management system, this activity defines a prioritized management plan to support the management system over the tactical time period of the IIS project.

A1422 ESTABLISH MANAGEMENT SYSTEM MONITORING PLAN

Using feedback from all processes, this activity assesses the effectiveness of the management system and makes sure that it is enhanced if necessary. Organizational performance is compared against management factors to determine if and how the management system should be modified to adequately meet the needs of the IIS evolution project.

A143 DEFINE SERVICE PLAN

This activity focuses on the development of a total service plan that will meet the needs of the enterprise over the tactical time period of the IIS evolution project. The service plan covers service marketing, service level planning, recovery planning, security planning, and audit planning.

A1431 ESTABLISH SERVICE MARKETING PLAN

Based upon the services to be offered during the tactical time period of the IIS evolution project, this activity establishes who can use the services and at what rate they are to be charged. Also during this activity, market prices and service volumes are forecast, and services promotions are developed.

A1432 ESTABLISH SERVICE LEVEL PLAN

Utilizing market prices and service volume forecasts, this activity establishes service agreements for the tactical time period of the IIS

evolution project. This includes: recovery, security, auditability, etc. A preventative maintenance plan and service support plan are also developed during this activity.

A1433 ESTABLISH RECOVERY PLAN

This activity establishes a plan to ensure that agreed upon services are provided in case of system failure.

A1434 ESTABLISH SECURITY PLAN

This activity establishes a plan to ensure that agreed upon levels of system and service security are provided.

A1435 ESTABLISH AUDIT PLAN

This activity establishes a plan to ensure that agreed upon levels of system and service auditability are provided.

A144 DEFINE RESOURCE PLAN

This activity focuses on integrating requirements for Development Planning and Service Planning into a workable resource plan for the IIS evolution project.

A1441 ESTABLISH CAPACITY PLAN

Utilizing workload forecasts from the IIS evolution project and/or from the evolution of existing services, this activity defines in a capacity plan how resources will cover the demand. It also purposes alternatives to management (number of shifts, decreased services, changes in systems plan, etc.). based upon the potential amount of total workload.

A1442 ESTABLISH SKILLS PLAN

Utilizing requirements identified in the IIS evolution project plan and from the service plan, this activity defines in the personnel and the education plans how existing and planned skills will meet the demand.

A1443 ESTABLISH BUDGET PLAN

This activity translates system, service, project, capacity, personnel, and education plans into financial terms and identifies how funds will be obtained and allocated during the IIS evolution project.

A15 MANAGE DEVELOPMENT OF TACTICAL PLANS

This activity merges individual plans and resolves any imbalances. It monitors performance against the total IIS strategy and determines any corrective action that is necessary.

A151 REVIEW PLANS

This activity focuses on the review of service, project, capacity, personnel, education, budget, system, application, and data plans related to the IIS evolution project.

A152 BALANCE PLANS

This activity focuses on balancing service, project, capacity, personnel, education, budget, system, application, and data plans related to the IIS evolution project.

A153 OBTAIN PRELIMINARY APPROVAL

This activity focuses on obtaining preliminary approval for service, project, capacity, personnel, education, budget, system, application, and data plans related to the IIS evolution project.

A154 PUBLISH PLANS

This activity focuses on publishing service, project, capacity, personnel, education, budget, system, application, and data plans that are related to the IIS evolution project.

A2 EFFECT IIS

This activity covers the administration of the IIS project; the evolution of the IIS; the provisions for a systems engineering environment to be used by the IIS; and the tooling, testware and documentation that is to be generated over the tactical time of the IIS evolution project.

A21 ADMINISTER IIS EVOLUTION PROJECT

This activity focuses on the management and control of the IIS evolution project.

A214 MANAGE EXECUTION OF IIS EVOLUTION PROJECT

A2141 CONTROL PROJECT

This activity focuses on tracking project progress against the tactical plan, reporting status, conducting project reviews, resolving problems, and submitting change requests.

A21411 PROVIDE PROJECT ASSIGNMENT

208

During this activity, tactical plans are utilized to establish the project scope, leadership, and user involvement necessary to ensure success of the IIS evolution project.

A21412 PROVIDE PROJECT SCHEDULING

During this activity a detailed IIS evolution project project plan is prepared. The plan includes: objectives, organization, resources, tasks, work schedule, and deliverables.

A21413 PROVIDE PROJECT MONITORING

Using the detailed project plan, this activity tracks the progress of the IIS evolution project project through its phases, conducting reviews, resolving development problems, and documenting a history of its life cycle. This includes submitting change requests to move deliverables to the system inventory.

A21414 PROVIDE PROJECT REQUIREMENTS CONTROL

This activity accepts or rejects requests for changes and adjusts the detailed IIS evolution project plan accordingly. Changes result from modified objectives, new procedures, new technology, or errors in the original project scope.

A21415 PROVIDE PROJECT EVALUATION

Utilizing the detailed IIS evolution project plan and project history, this activity documents its completion and ensures that all promised deliverables were actually furnished. It compares actual and planned results and identifies reasons for variances.

A2142 CONTROL EVOLUTION

This activity maintains the evolution of the IIS through control of application/software development, procurement and upgrade; control of hardware/facility installation and upgrade; and tuning/system balancing.

A21421 PROVIDE APPLICATION/SOFTWARE DEVELOPMENT AND UPGRADE

This activity focuses on the development and upgrade of applications, operating systems, and other support software that is utilized by the IIS.

A21422 PROVIDE APPLICATION/SOFTWARE PROCUREMENT AND UPGRADE

This activity focuses on the procurement of vendor developed applications, operating systems, and other support software to be modified and upgraded for use by the IIS.

A21423 HARDWARE/FACILITY INSTALLATION AND UPGRADE

This activity focuses on the selection, installation, removal, modification and upgrade of IMS hardware and facilities that are to be utilized by the IIS.

A21424 PROVIDE TUNING AND SYSTEM BALANCING

This activity tunes resources to ensure that the IIS evolution proceeds as desired. This includes periodic evaluation to make sure that modifications to the evolution do not cause undesirable results in existing resources and so that system balancing can take place to rectify problems.

A2143 CONTROL RESOURCES

This activity maintains the flow of IIS project deliverables and controls changes the project resources resulting therefrom.

A21431 PROVIDE RESOURCE CHANGE CONTROL

Using change requests, this activity selects, coordinates, groups, and monitors all changes to IIS evolution project resources and procedures in such a way that there is either minimal impact on operations or minimal risk. It triggers resource and data inventory updates.

A21432 PROVIDE RESOURCE AND DATA INVENTORY CONTROL

Using change information, this activity builds and manages inventories of all the IIS evolution project resources (including personnel and financial).

A2144 CONTROL SERVICES

This activity maintains production and distribution of services related to the IIS evolution project.

A21441 PROVIDE PRODUCTION AND DISTRIBUTION SCHEDULING

Based upon tactical plans and inventory related to the IIS evolution project, this activity translates planned service agreements into a

work schedule for production and distribution. This activity also monitors work performance and if deemed necessary modifies the the schedule accordingly.

A21442 PROVIDE RESOURCE AND DATA PERFORMANCE CONTROL

Based upon the work schedule related to the IIS evolution project, this activity measures, quantifies and reports system performance levels. If a problem situation occurs, a pre-defined corrective action is taken to adjusted work tasks in order to enhance system performance. If no pre-defined corrective action is available the PROVIDE PROBLEM CONTROL activity is initiated.

A21443 PROVIDE PROBLEM CONTROL

This activity determines the nature, impact and extent of problems related to the IIS evolution project. Once a problem has been recognized and logged, bypass and recovery procedures are activated to resolve the problem. This activity also controls and reports the status of all problems that are being resolved.

A21444 PROVIDE SERVICE EVALUATION

This activity compares performance and impact of problems with service agreements. It also identifies and reports reasons for variances to users and management.

A2145 PROVIDE ADMINISTRATIVE SERVICES

This activity administers the financial, personnel, education, and training activities that are required to support the IIS evolution project.

A21451 PROVIDE FINANCIAL ADMINISTRATION

This activity applies service rates to resources to determine total costs applicable to individual users, accumulates these costs, charges the individual user organization, and matches against the budget. This activity also includes maintaining the contractual agreements for project work, hardware and software leasing, and other supportive efforts.

A21452 PROVIDE STAFF PERFORMANCE TRACKING

This activity tracks staff performance and reports productivity associated with the IIS evolution project.

A21453 PROVIDE EDUCATION AND TRAINING

This activity educates users and personnel on IIS-related topics through job training or formal education.

A2146 PROVIDE INFORMATION MANAGEMENT SERVICES (IMS)

This activity interprets and executes required activities in order to deliver production, distribution, and supporting services that are related to the IIS evolution project.

• A21461 PROVIDE IMS FOR PRODUCTION

This activity receives, schedules, executes and makes available for distribution: jobs, transactions, and data to satisfy service agreements. It monitors work progress against the production and distribution schedule and takes corrective action when problems occur. This activity also executes pre-defined or emergency procedures for bypass, recovery, security, and/or performance.

A21462 PROVIDE IMS FOR DISTRIBUTION

This activity receives, stores and sends information and data through the distribution network to satisfy service agreements. It monitors information movement against the production and distribution schedule and takes corrective action should problems occur. This activity also executes pre-defined or emergency procedures for bypass, recovery, security, and/or performance.

A21463 PROVIDE IMS FOR CUSTOMER SERVICE

This activity provides an interface that enables a better understanding of customer expectations. It also provides customer support, whereby help is offered by the appropriate service. Problems are forwarded to the appropriate function.

A21464 PROVIDE IMS FOR SERVICE MARKETING

This activity markets services to customers and identifies needs for future services. This activity also initiates actions to provide services.

A22 EVOLVE IIS

This activity focuses on the evolutionary steps that are undertaken to develop an operational IIS.

APPENDIX C

IDS/SDPO MODEL

IDS/SDP-0 MODEL

213

IDS/SDP-0 MODEL

(INTEGRATED DESIGN SYSTEM/SYSTEM DEVELOPMENT

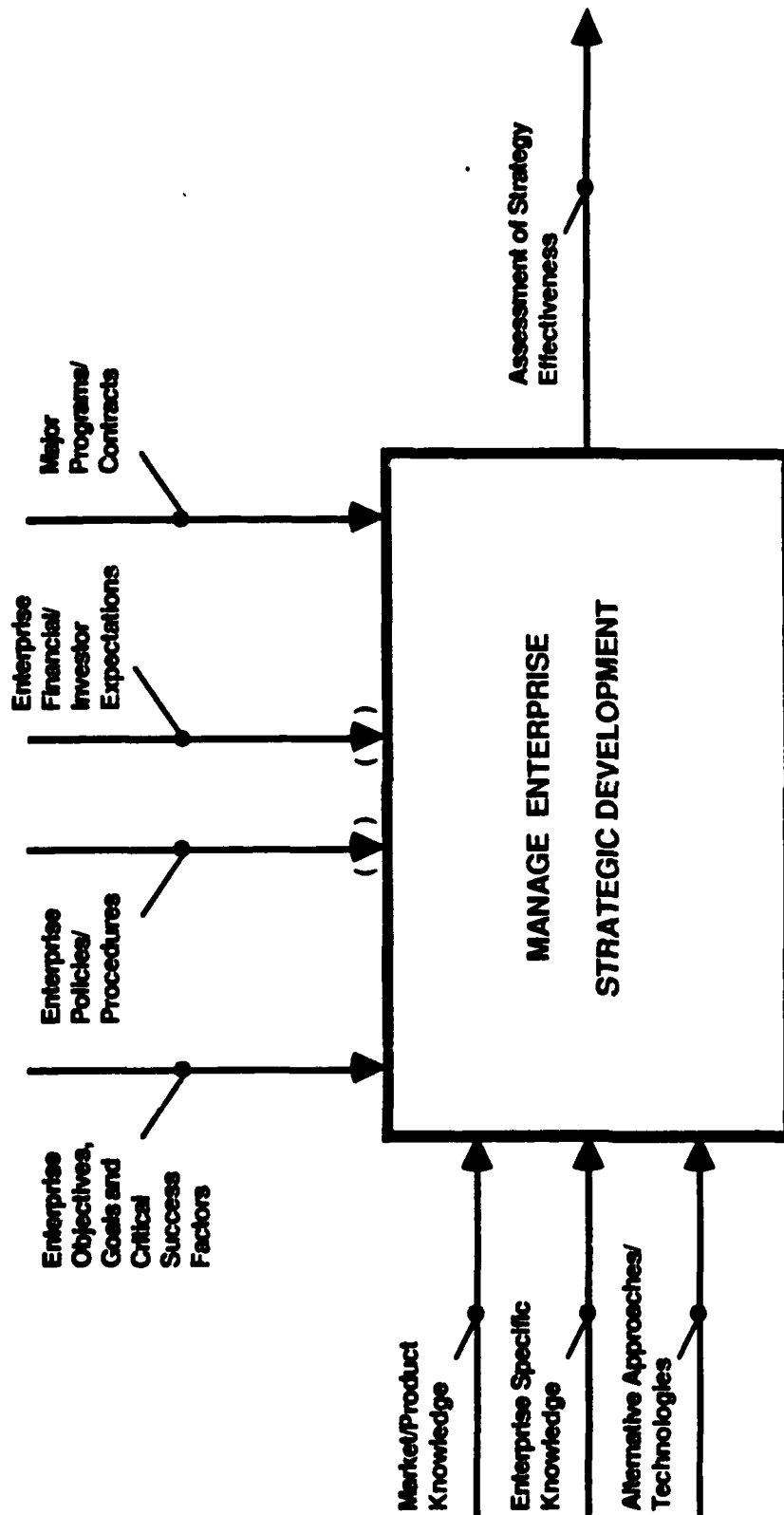
PROCESS IDEF-0 MODEL)

PURPOSE: TO REPRESENT AN ACTIVITY FRAMEWORK WHICH CAN BE USED AS A SKELETON FOR PLANNING HOW TO EVALUATE, CUSTOMIZE AND UTILIZE (ASSIMILATE) THE TECHNOLOGY AND CAPABILITY COMPRISING AN IDS SHELL.

VIEWPOINT: THE PLANNER/MANAGER RESPONSIBLE FOR INTRODUCTION OF IDS CAPABILITY INTO THE ENTERPRISE, AND THE USE OF THIS CAPABILITY IN THE BEST INTERESTS OF THE ENTERPRISE AS A WHOLE.

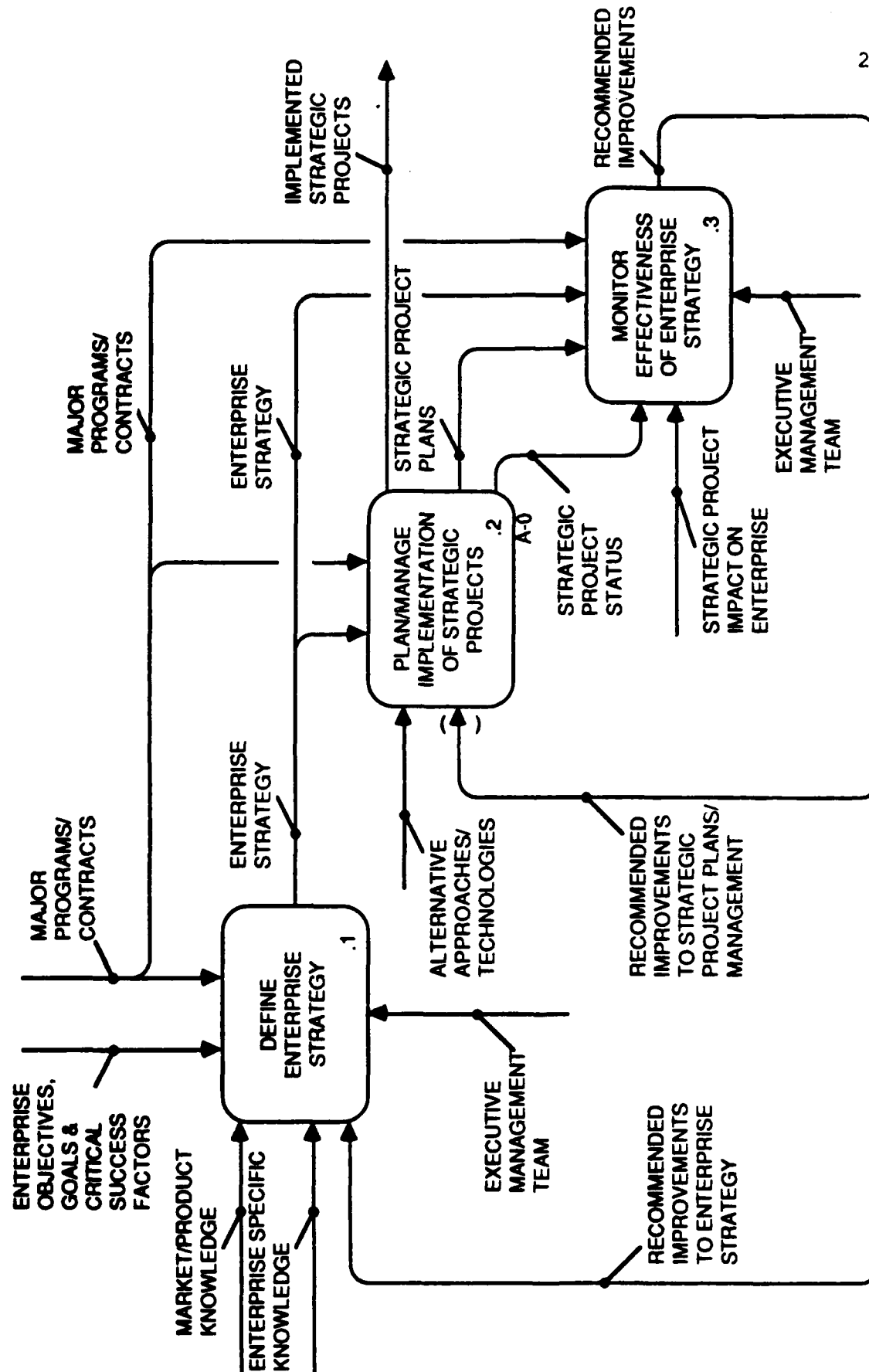
A-1 MANAGE ENTERPRISE STRATEGIC DEVELOPMENT

The very nature of information integration vehicles such as the IDS implies strategic ramifications to an Enterprise. While it is certainly true that IDS-like capabilities can be employed on a much more limited scale than as an Enterprise-wide integration catalyst, experience to date with such devices appears to indicate that the greatest leverage might be gained from their employment throughout the Enterprise as a whole, as opposed to program/project confined utilization. This might, of course, require significant advances in technology over what is commonly (and affordably) available today. However, with their strategic potential having already been demonstrated, and with the requisite advances in technology apparently "in the wings" so to speak, the decision to employ IDS-like vehicles is perceived as being justifiably strategic in nature, and the effort to implement one is likely to be viewed as another of several on-going strategic projects in most Enterprises in the future. The IDS/SDP-0 model attempts to address the implementation and use of an Information Integration shell, and an IDS shell in particular, from this strategic perspective.



A-1

IDS/SDP-0 CONTEXT OVERVIEW (FEO)

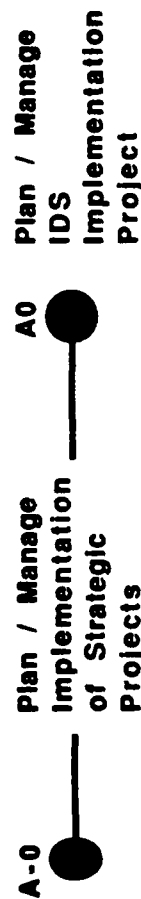


A-1 (CONTEXT): MANAGE ENTERPRISE STRATEGIC DEVELOPMENT (FEO)

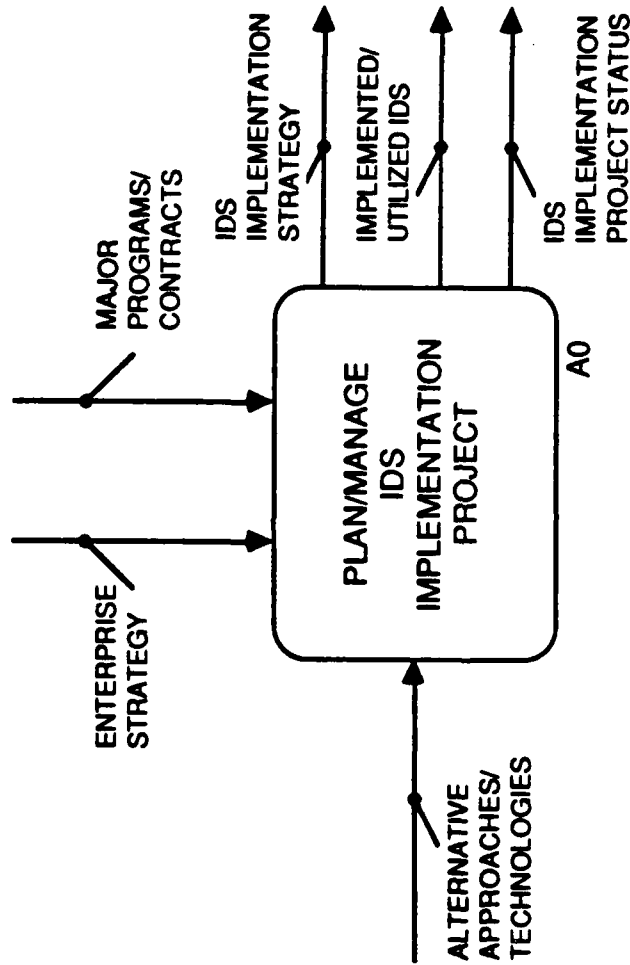
A-0 PLAN/MANAGE IMPLEMENTATION OF STRATEGIC PROJECTS

Strategic projects are generally recognized as long-term efforts which contribute to strengthening the position of an Enterprise with respect to its market and/or competition. They tend to be "sold" to Executives based more on their potential benefits than on assurances of cost/benefit relationships, since often times neither the total cost of a strategic project nor its implementation schedule can be projected with any appreciable degree of certainty. Funding is generally provided on an incremental basis, predicated on a continuous stream of positive indications to Enterprise Executives that the effort warrants additional investment.

Implementation of an IDS-like capability is likely to be viewed by Executives as a strategic effort; one of several that the Executive body of the Enterprise is pursuing. As such it is likely to be directed and funded in a manner consistent with the handling of other strategic projects in the Enterprise. The IDS/SDP-0 model reflects this assumption by treating an IDS implementation project as one "instance" of the strategic project portfolio of the Enterprise.



A-0 CONTEXT OVERVIEW NODE TREE (FEO)



A-0 (CONTEXT): PLAN/MANAGE IMPLEMENTATION OF STRATEGIC PROJECTS (FEO)

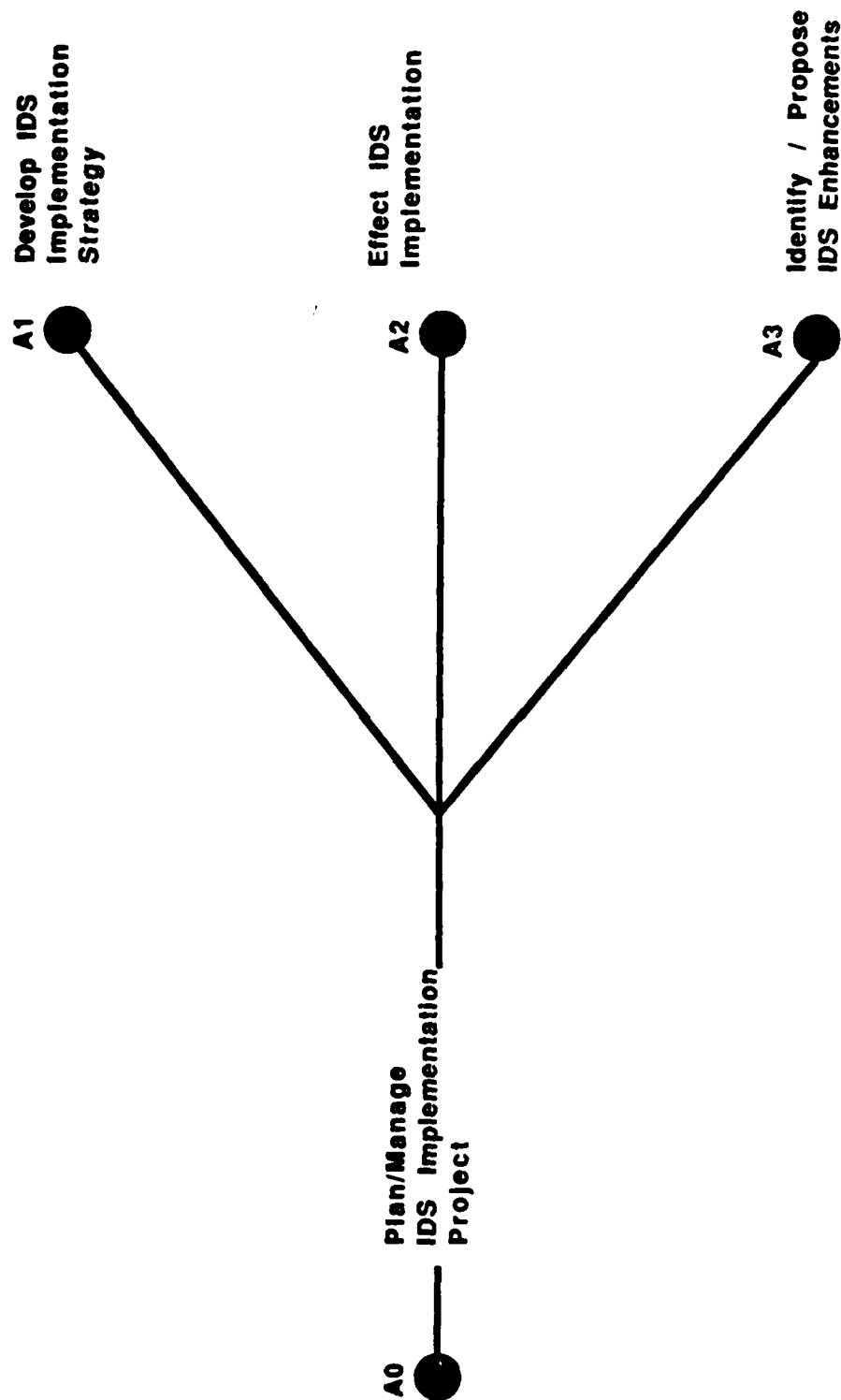
A0 PLAN/MANAGE IDS IMPLEMENTATION PROJECT

In the broadest of terms, the implementation of an IDS-like capability in an Enterprise is expected to track closely with the traditional life-cycle based administrative view of major projects, to the extent depicted below:

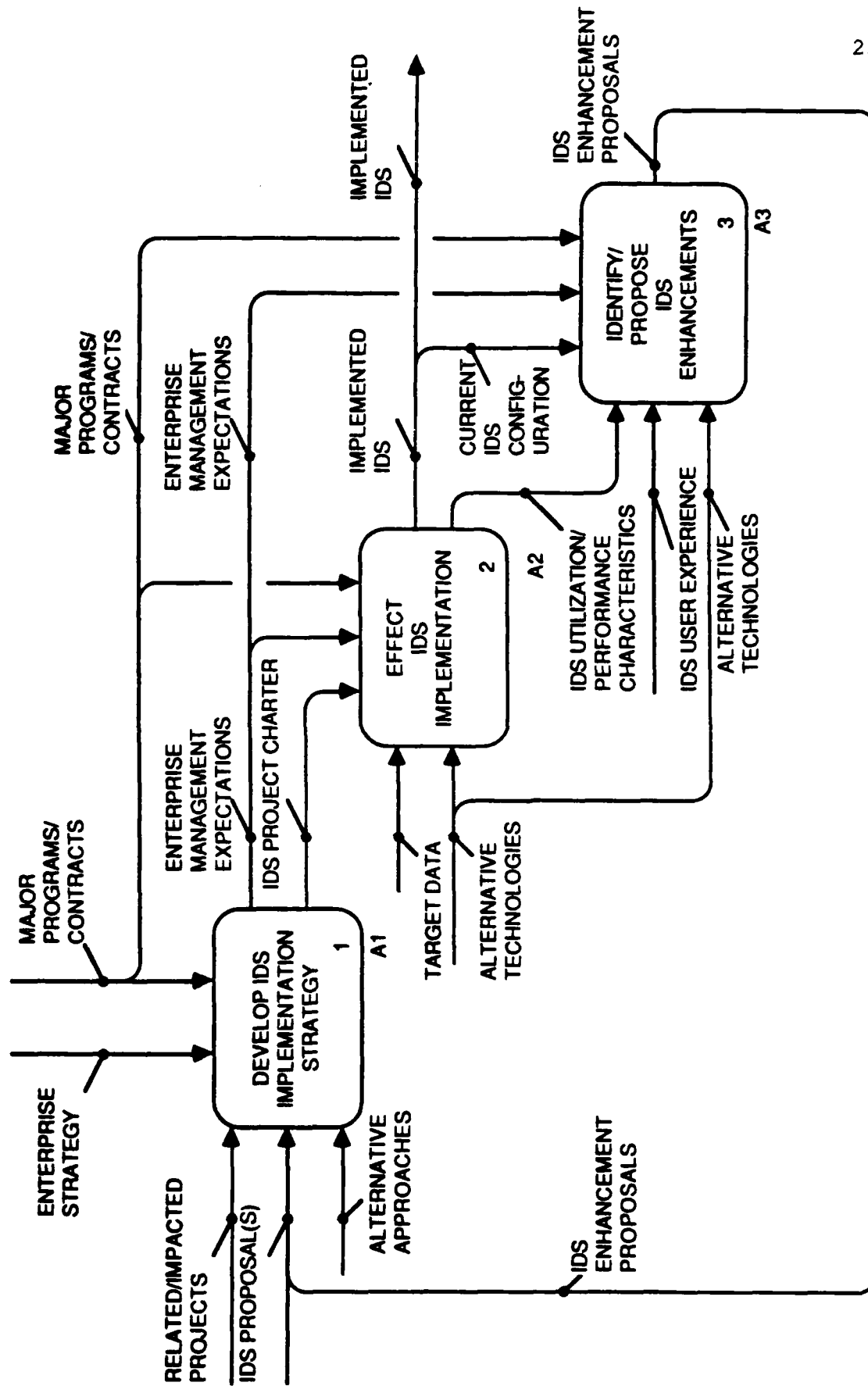
- o Understand the problem
- o Implement a solution
- o Improve the solution

However, several significant departures from tradition can be noted at the more detail levels of the IDS/SDP-0 model, particularly in the technical portions (A22 and A23). This is primarily due to the perception of data as a shareable resource that is employable by many computer applications, and by the use of recursive and prototype-based approaches to the development and use of IDS-like capabilities.

Additionally, the IDS/SDP-0 model reflects the view that an IDS (or IDS-like) facility is really an enabling technology directed at achieving information integration in complex computerized environments, and that the environment would gradually (and at its own pace) evolve in the direction of integration, not achieve it overnight through one large technology revolution. As a result, the IDS/SDP-0 model assumes continued refinement/tuning of the IDS technology itself during this evolutionary process.



A 0 IDS/SDP-0 NODE TREE (FEO)

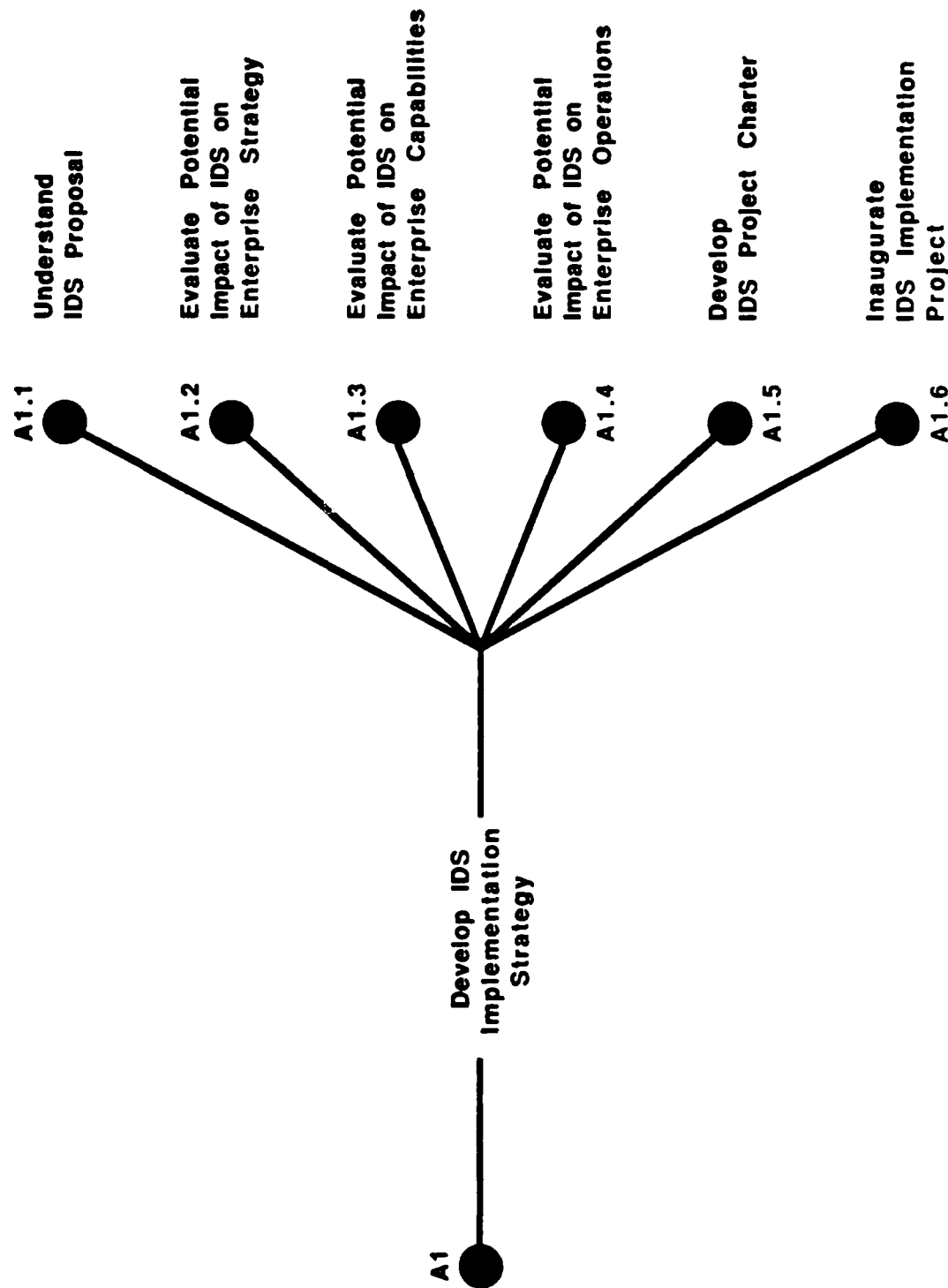


AO PLAN/MANAGE IDS IMPLEMENTATION PROJECT

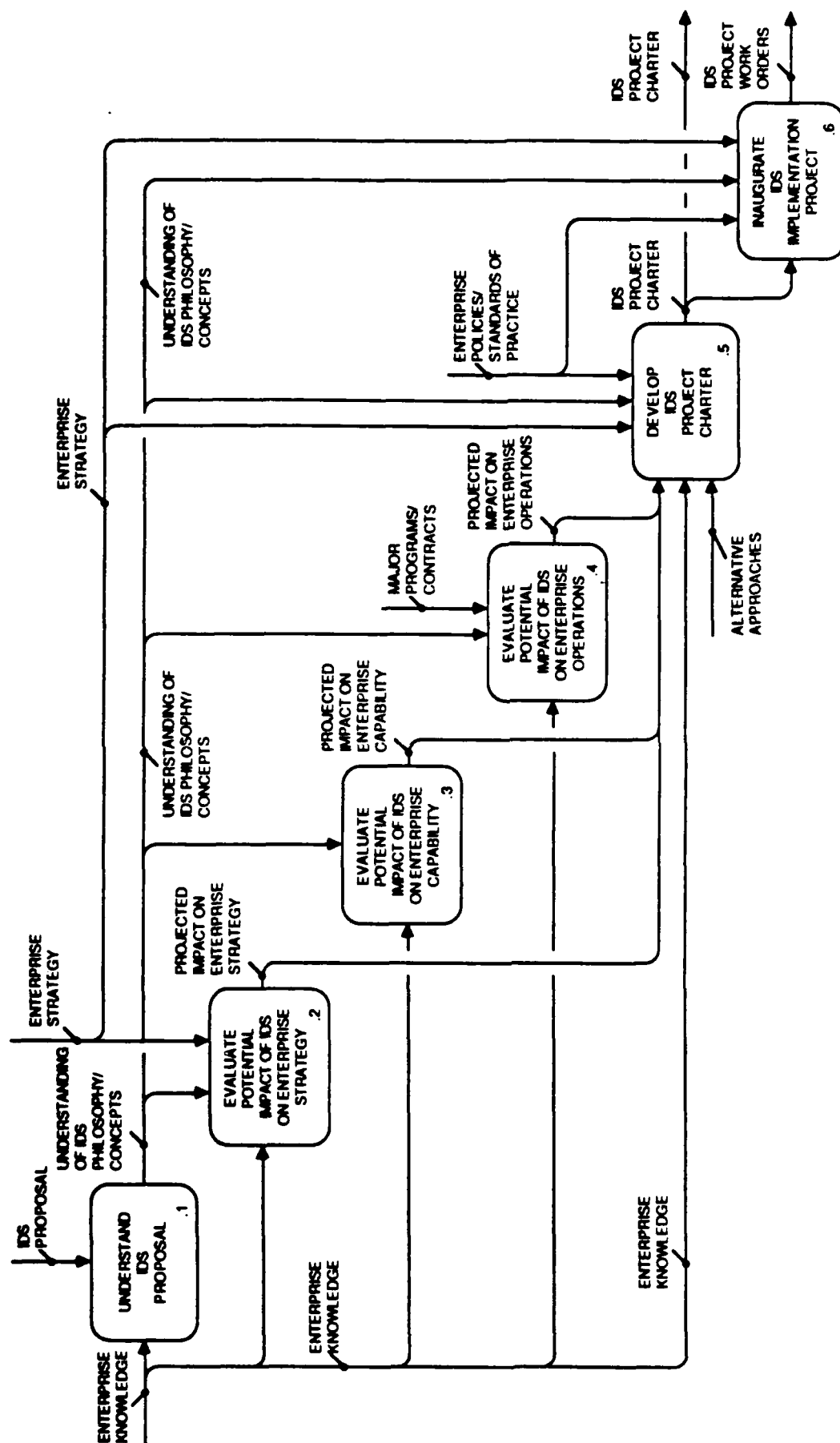
A1 DEVELOP IDS IMPLEMENTATION STRATEGY

Prior to approving/sponsoring an IDS-like implementation effort, the Enterprise's Executive management should understand the ramifications of having such a capability. Based on an informed understanding of the IDS concept and intent, Executive management should appraise the potential impact of IDS capability on their own Enterprise. Predicated on this assessment, objectives, goals, critical success factors and other expectations can be established for the IDS implementation project. In addition, appropriate constraints can be outlined, e.g. limitations on initial funding levels, milestone qualifications, which will contain any perceived risk within tolerable limits.

If an IDS shell (or similar shell) is to be employed as the core of the desired capability, the IDS/SDP-0 itself can be used as the framework for defining, in broad terms, the desired approach; it becomes both a guide and checklist for management planning. Once the general, long-range approach is determined, a Strategic Project Management team should be established, an Executive Sponsor clearly identified, and the Project Manager authorized to proceed.



A1 NODE TREE (FEO)



A1 DEVELOP IDS IMPLEMENTATION STRATEGY

A1 DETAIL NODE INDEX

- A1 Develop IDS Implementation Strategy
 - A1.1 Understand IDS Proposal
 - A1.2 Evaluate Potential Impact of IDS on Enterprise Strategy
 - A12.1 Examine Potential Impact on Enterprise Mission
 - A12.2 Examine Potential Impact on Enterprise Products
 - A12.3 Examine Potential Impact on Enterprise Policies
 - A12.4 Examine Potential Impact on Enterprise Economics
 - A12.5 Examine Potential Impact on Enterprise Objectives/Goals
 - A12.6 Examine Potential Impact on Enterprise Critical Success Factors (CSF's)
 - A1.3 Evaluate Potential Impact of IDS on Enterprise Capabilities
 - A13.1 Examine Potential Impact on Marketing Capability
 - A13.2 Examine Potential Impact on Production Capability
 - A13.3 Examine Potential Impact on Service Capability

- A1.4 Examine Potential Impact of IDS on Enterprise Operations
 - A14.1 Examine Potential Impact on Enterprise Function Management
 - A14.2 Examine Potential Impact on Enterprise Information Management
 - A14.3 Examine Potential Impact on Enterprise Technology Base
 - A14.4 Examine Potential Impact on Enterprise Skill Base
 - A14.5 Examine Potential Impact on Known Operational Problems
 - A14.6 Examine Potential Impact on Other Enterprise Projects
- A1.5 Develop IDS Project Charter
 - A15.1 Define IDS Project Objectives/Goals
 - A15.2 Define IDS Project Critical Success Factors (CSF's)
 - A15.3 Define IDS Project Constraints
 - A15.4 Define IDS Implementation Approach
 - A15.5 Structure IDS Project Management Team
- A1.6 Inaugurate IDS Implementation Project
 - A16.1 Acquire Approval of IDS Project Charter
 - A16.2 Acquire Initial Funding Authorization
 - A16.3 Assign IDS Project Management Team Members
 - A16.4 Release IDS Project Work Orders

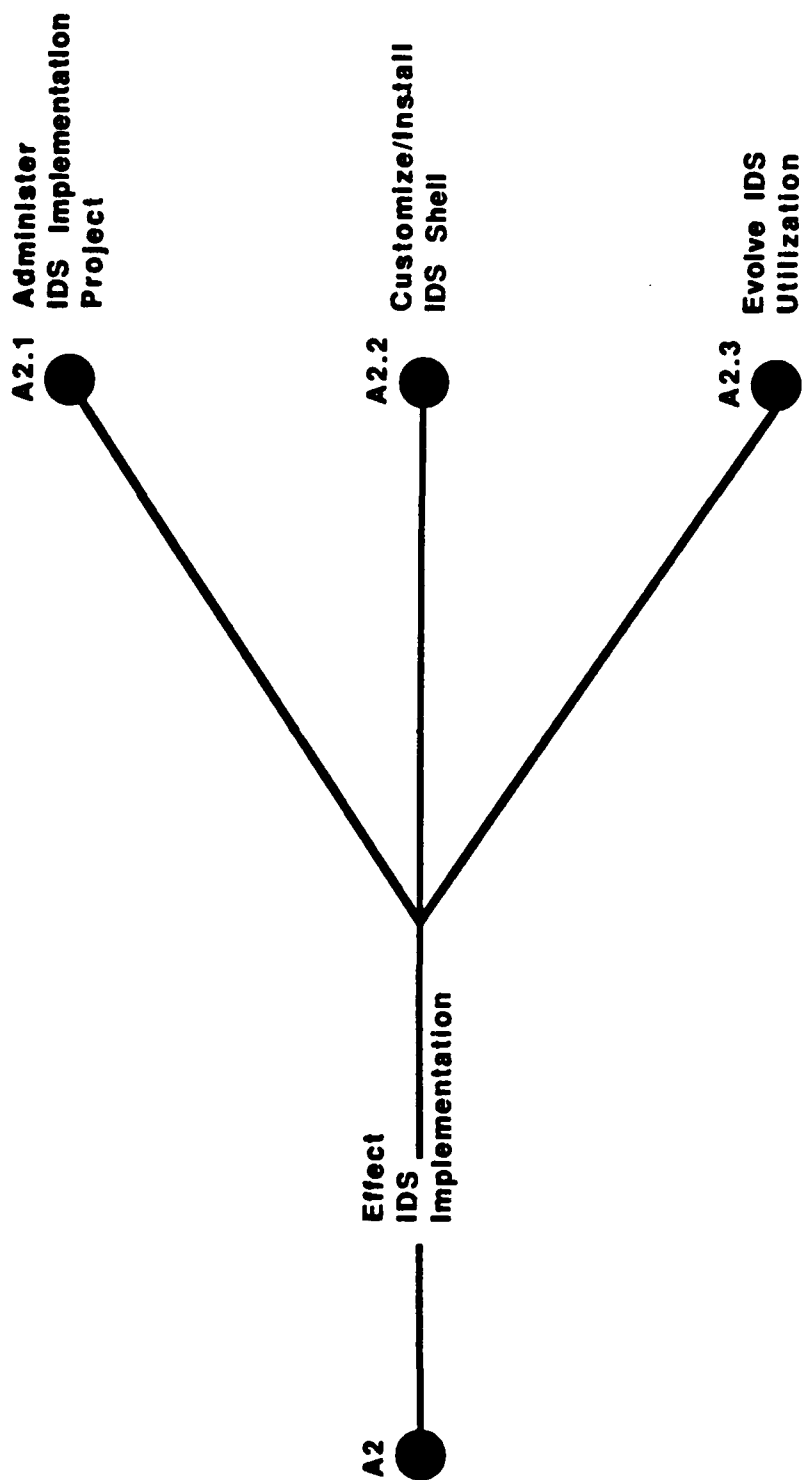
A2 EFFECT IDS IMPLEMENTATION

Implementation of an IDS shell is much more evolutionary than we perceive traditional application development to be, and far more complex. It is anticipated that this evolution will continue for several years, perhaps a decade or more in large, complex environments. The capabilities of an IDS-like facility are also envisioned to necessitate significant change of the way an Enterprise manages/uses its data, as well as in how it goes about the business of constructing its information systems. Perhaps the greatest challenge to the Enterprise will be to positively embrace these changes, and the greatest challenge to the IDS Project Management Team will be to manage the rate and impact of such changes.

The approach outlined in this portion of the IDS/SDP-0 model breaks the implementation problem into three distinct (but inter-related) components:

- o Planning and Managing the effort
- o Customizing/installing an IDS (or IDS-like) shell
- o Evolving the use of IDS capabilities

The first of these encompasses the Administrative tasks to be addressed in an effort of this type, while the last two address technically oriented tasks.



A2 NODE TREE (FEO)

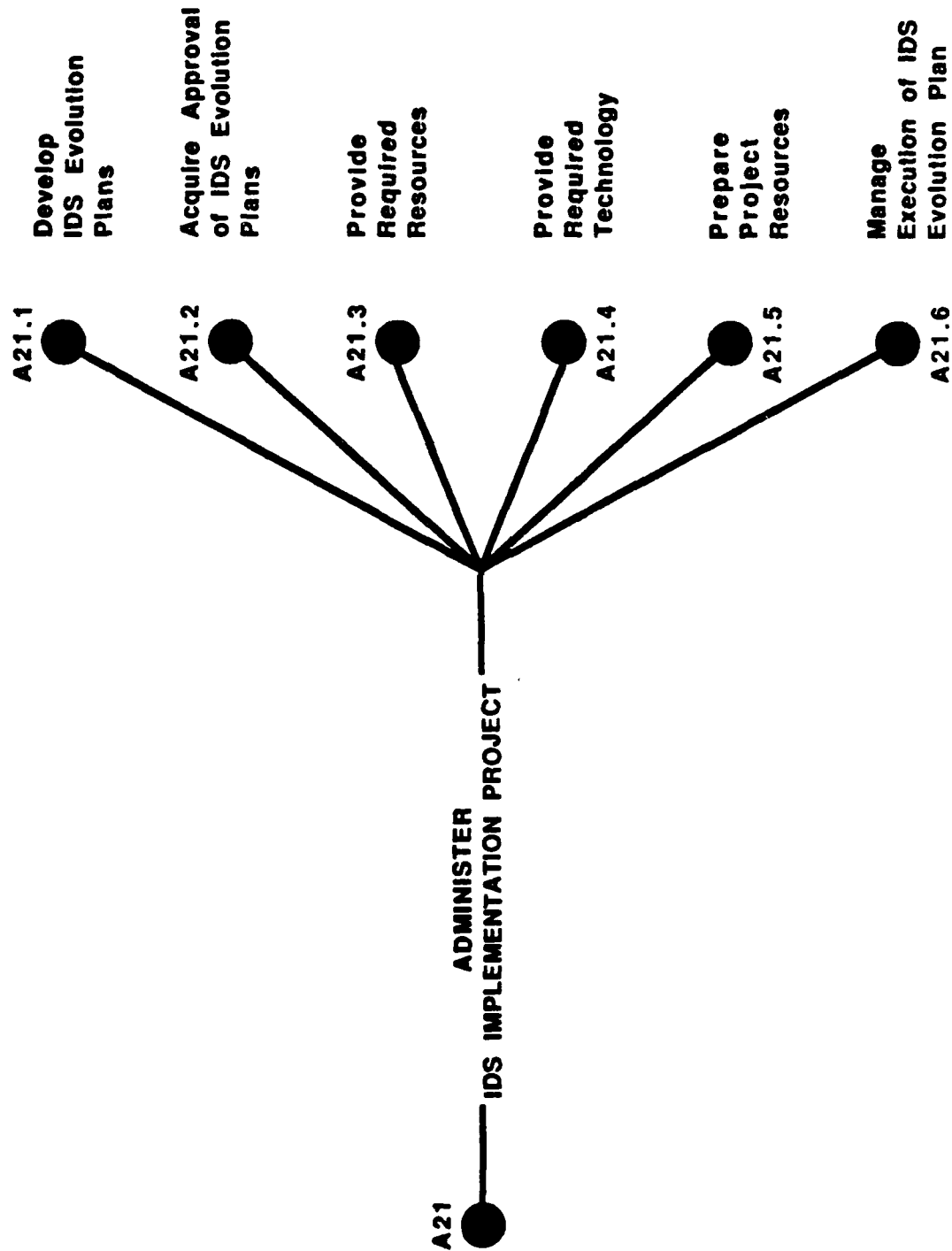


A2 EFFECT IDS IMPLEMENTATION

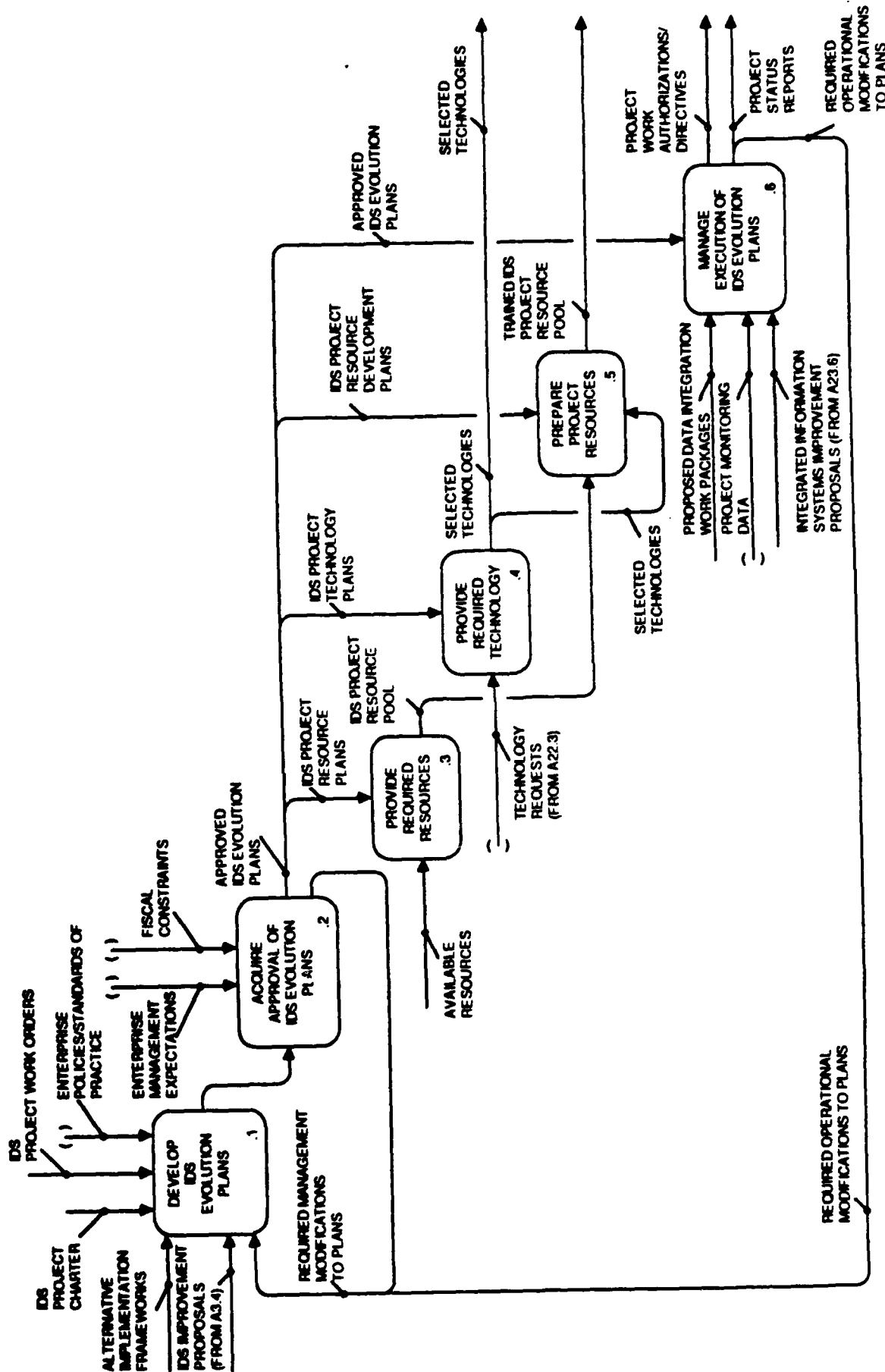
A21 ADMINISTER IDS IMPLEMENTATION PROJECT

Due to the strategic nature of an IDS Implementation project, it is anticipated that the administrative functions of the project will be cyclical. The first cycle will most likely be directed at what is required to customize and install the IDS shell as the first step in the IDS evolution process. Subsequent administrative cycles will be targeted at gradually expanding the purview of the IDS capabilities.

The IDS/SDP-0 model is intended to provide a framework which can be tailored to become an Enterprise-specific plan for IDS implementation. The administrative function is responsible for this tailoring and planning. The administrative function is also responsible for assuring that human resources are adequately prepared for the kinds of changes that IDS capability is expected to bring about, i.e., for managing the potential culture shock that comes with significant change. Additionally, this function assures that the requisite resources and technology are provided to support the IDS effort.



A21 NODE TREE (FEO)



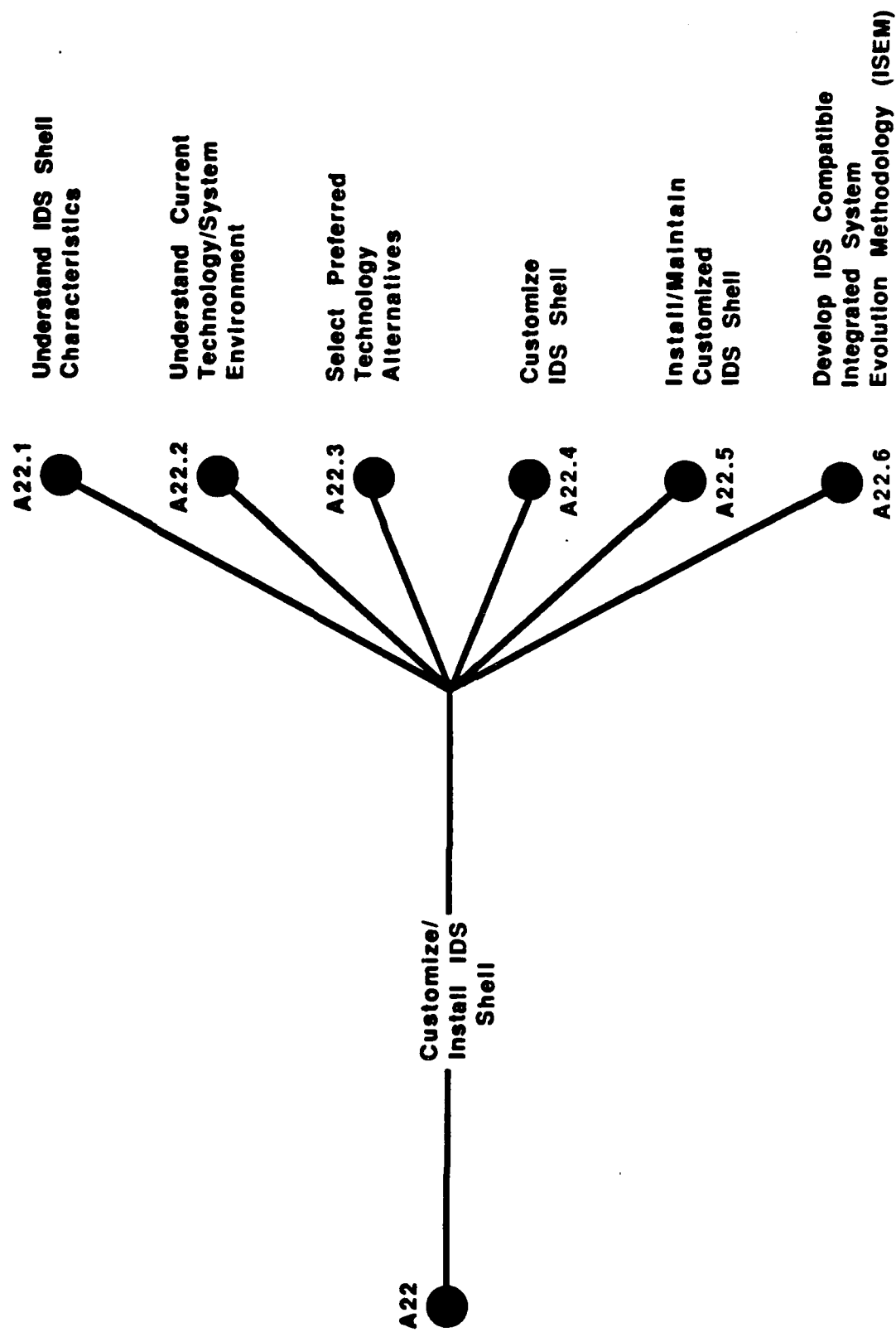
A21 ADMINISTER IDS IMPLEMENTATION PROJECT

A22 CUSTOMIZE/INSTALL IDS SHELL

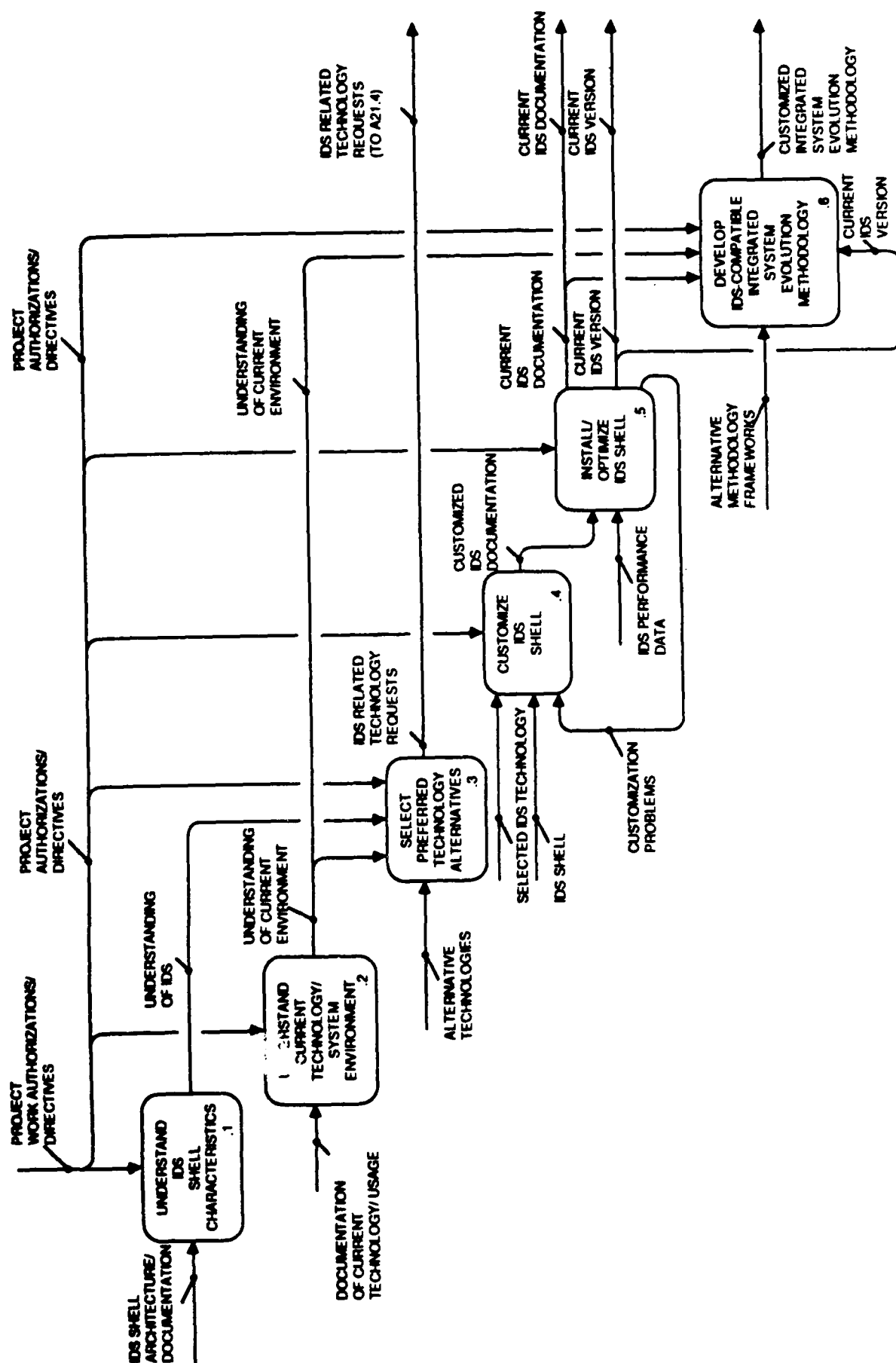
An IDS shell is anticipated to be similar, in some respects, to an expert system shell. Its embedded expertise is directed at managing and facilitating the optimum use of shareable data. It can be viewed as an enabling technology through which integration of information systems is achieved and managed.

Although an IDS shell is expected to include most of the technology required to enable it, certain "Hooks" will likely be provided which will enable the Enterprise to connect the IDS with its already installed technology base. In addition, some of the functionality of an IDS can potentially be provided by more than one technology, and the Enterprise may wish to replace some of the natural technology components of the IDS shell with their own.

Last, but by no means least, is the need to define a system development/evolution methodology which is compatible with the functionality of the installed IDS. While possibly based on the IDS/SDP-0 model, this will likely be a blend of IDS-dependent approaches and some vestiges of their current approaches being employed in the Enterprise.



A22 **NODE TREE (FEO)**



A22 CUSTOMIZE/INSTALL IDS SHELL

A226 NODE INDEX

- A226 Develop IDS Compatible Integrated System Evolution Methodology
 - A226.1 Understand Impact of Current IDS on Alternative Methodology Frameworks
 - A226.2 Construct Alternative Scenarios for using IDS Capabilities
 - A226.3 Conduct Pilot Exercise(s) to Validate Scenario Assumptions
 - A226.4 Select/Customize Preferred Methodology Framework(s)
 - A226.5 Document Customized Integrated System Evolution Methodology

A23 EVOLVE IDS UTILIZATION

This is perhaps the most complex of the IDS implementation processes. Although separated functionally from the customization/installation of the IDS shell, it is entirely feasible that this function will overlap significantly the same time frame.

The first process is directed at identification and formulation of data integration work packages. Each of these work packages is comprised of an information boundary and an activity boundary (in terms of Enterprise activities). They are anticipated to be prioritized based on the degree to which the Enterprise is dependent on/sensitive to the various kinds of information it consumes/produces. The first work package released should address the kind of information the Enterprise feels will have the most positive impact in the shortest period of time when standardized and managed via the IDS (or IDS-like facility). Subsequent work package releases would also be based on the relative impact of their information content on the Enterprise.

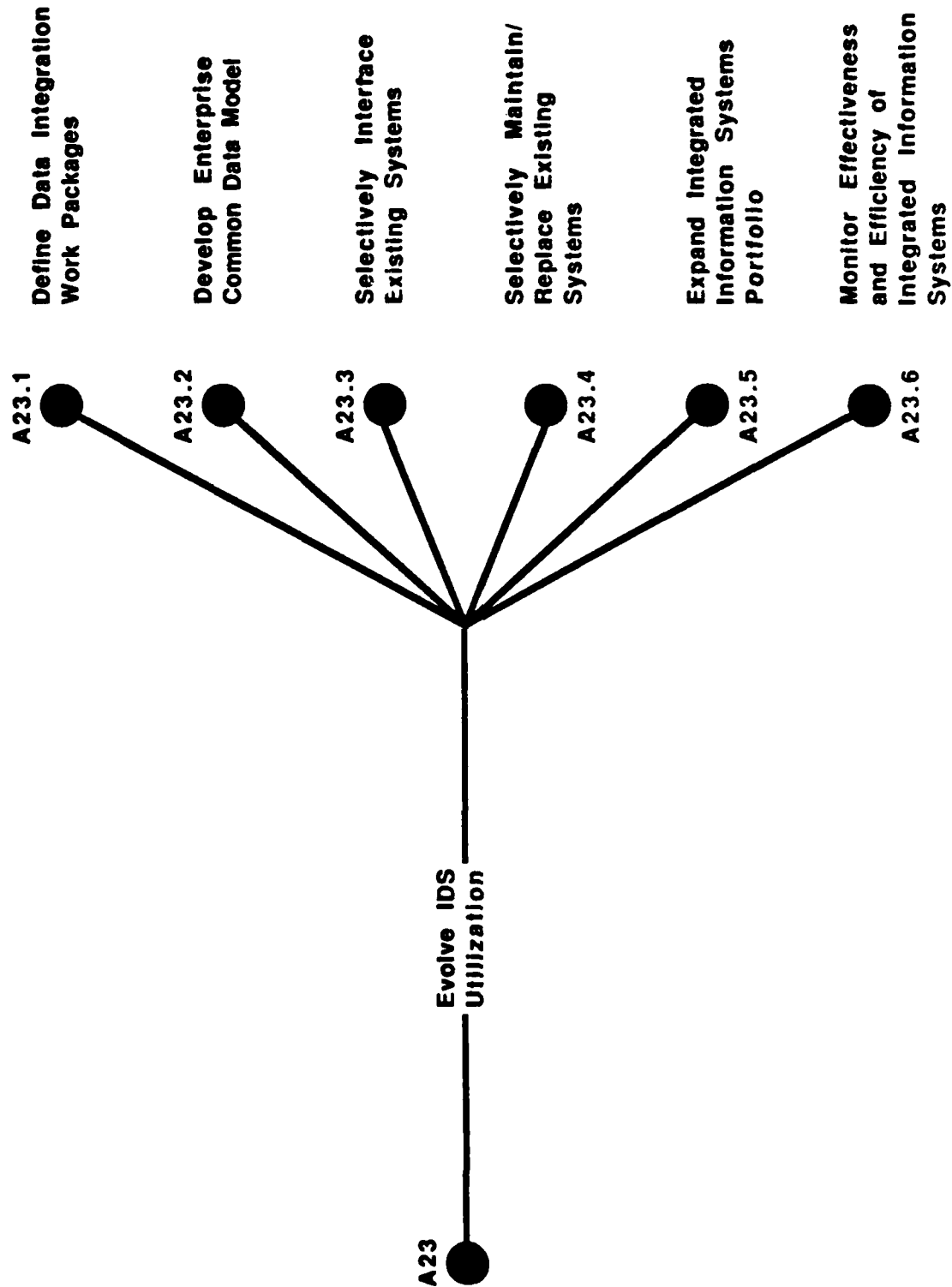
A fundamental step in the direction of information integration is the development of a common data model. An information model should be constructed for each data integration work package, capturing an Enterprise-wide (common) view of the data comprising the information boundary of the work package. As each work package information model is completed, it is integrated with any previous information models to gradually evolve an Enterprise Common Data Model in the IDS Knowledge Base.

There appear to be three major ways in which to employ the evolving Enterprise Common Data Model. Since it represents a common standard for shareable data, one way to use it is to help interface existing application databases via the IDS. It is highly unlikely that existing data, much of it redundantly appropriated in single/special purpose files or databases, will conform to the new Enterprise data standard. Some may be "within tolerance", i.e. useful as is without significant risk of misinterpretation. However, some will likely be subjected to "retrofit" to bring them into such tolerance. This will facilitate the use of data from multiple databases/files to formulate information previously inaccessible to the Enterprise. This is perhaps the fastest way in which to get a positive return on the IDS investment, but by no means the place to stop.

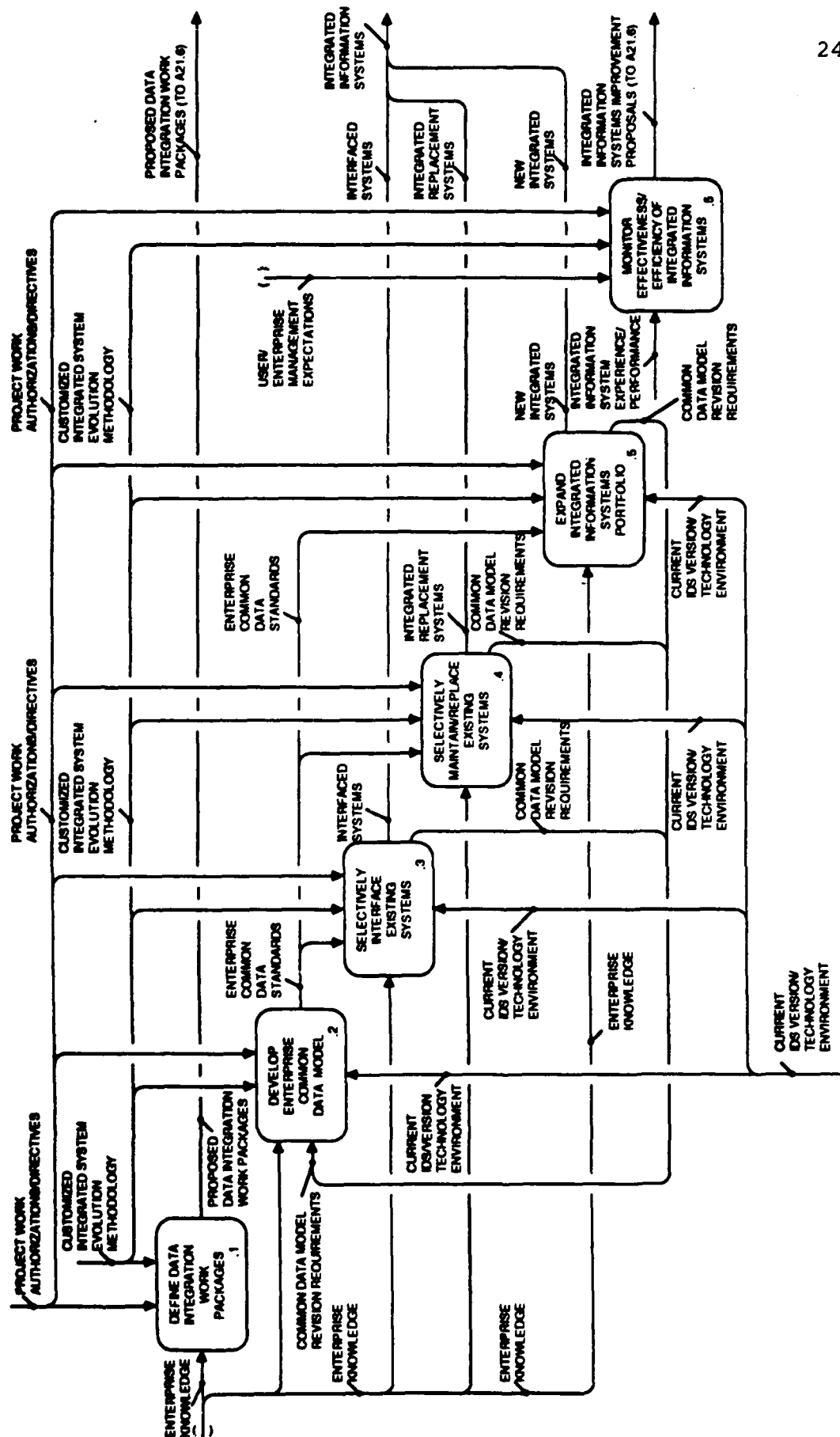
A second way of employing the IDS capability is to gradually reduce redundant data acquisition by selectively replacing some of the existing databases with new, shared databases that serve multiple applications. The new databases would conform to the Enterprise data standards in the IDS, thereby facilitating even broader potential use of the data than just the applications in the data originally serviced. A step in this direction is to gradually increase the conformance level of existing databases during the performance of system maintenance. As the conformance level increases, replacement will gradually become a practical and cost effective alternative.

A third way of employing the IDS capability is to add new databases which act as data acquisition vehicles for data not previously available in automated form. These new databases would also conform to the Enterprise data standards, thereby providing for additional uses of data than perhaps foreseen when the databases were first conceived.

Because of the kind of technology comprising the IDS facility, and its strong inclination towards managing data as an Enterprise resource rather than an application resource, it becomes feasible to address an additional problem statistically associated with traditional systems development approaches; correctness/completeness of user requirements. The IDS technology lends itself to the use of prototyping techniques to discover and demonstrate complete/correct information system requirements in a cost effective manner. Additionally, as time progresses and the IDS implementation occurs, it is highly likely that more emphasis will be placed on discovering optimum uses of data than on acquiring new kinds of data. Ultimately the Enterprise should evolve toward standard structures for acquiring and maintaining data, i.e. an Enterprise Data Acquisition System, and specialized ways of assembling data into useful information, i.e. Data Application Systems.



A23 NODE TREE (FEO)



A23 EVOLVE IDS UTILIZATION

A231 DETAIL NODE INDEX

- A231 Define Data Integration Work Packages
 - A231.1 Understand "As Is" Environment
 - A231.2 Identify/Understand Enterprise CSF Inhibitors
 - A231.3 Identify/Assess Alternative Inhibitor Neutralization Approaches
 - A231.4 Define "To Be" Environment
 - A231.5 Determine Information Priorities
 - A231.6 Identify/Prioritize Data Integration Work Packages

A232 DETAIL NODE INDEX

- A232 Develop Enterprise Common Data Model
 - A232.1 Construct Enterprise Information Model for Each Work Package
 - A232.2 Validate Enterprise Information Model for Each Work Package
 - A232.3 Integrate Enterprise Information Models From All Work Packages
 - A232.4 Identify/Resolve Conflicts Among Integrated Information Models
 - A232.5 Formulate Enterprise Common Data Model
 - A232.6 Populate IDS Knowledge Base (Conceptual Schema Definition)

A233 DETAIL NODE INDEX

- A233 Selectively Interface Existing Systems
 - A233.1 Establish Mapping Between Enterprise Common Data Model and Selected Database(s)
 - A233.2 Populate IDS Knowledge Base (Internal/External Schema Definitions)
 - A233.3 Construct/Operate Interface Validation Prototype(s)
 - A233.4 Retrofit Interfaced Database(s) as Practical
 - A233.5 Stabilize IDS Interfacing Facilities

A234 DETAIL NODE INDEX

- A234 Selectively Maintain/Replace Existing Systems
 - A234.1 Provide Shared Database/Data Acquisition Sub-System
 - A234.2 Populate IDS Knowledge Base (Internal/External Schema Definitions)
 - A234.3 Operate System Prototype(s)
 - A234.4 Retrofit/Replace Affected Data Application Sub-System Components
 - A234.5 Stabilize Replacement System
 - A234.6 Deactivate Vestigial System Components

A235 DETAIL NODE INDEX

A235 Expand Integrated Information Systems Portfolio

A235.1 Understand New System Requirements

A235.2 Provide New Shared Database/Data Acquisition Sub-System
(Internal/External Schema Definitions)

A235.3 Populate IDS Knowledge Base

A235.4 Provide New Data Application Sub-System (External Schema Definitions)

A235.5 Operate New System Validation Prototype(s)

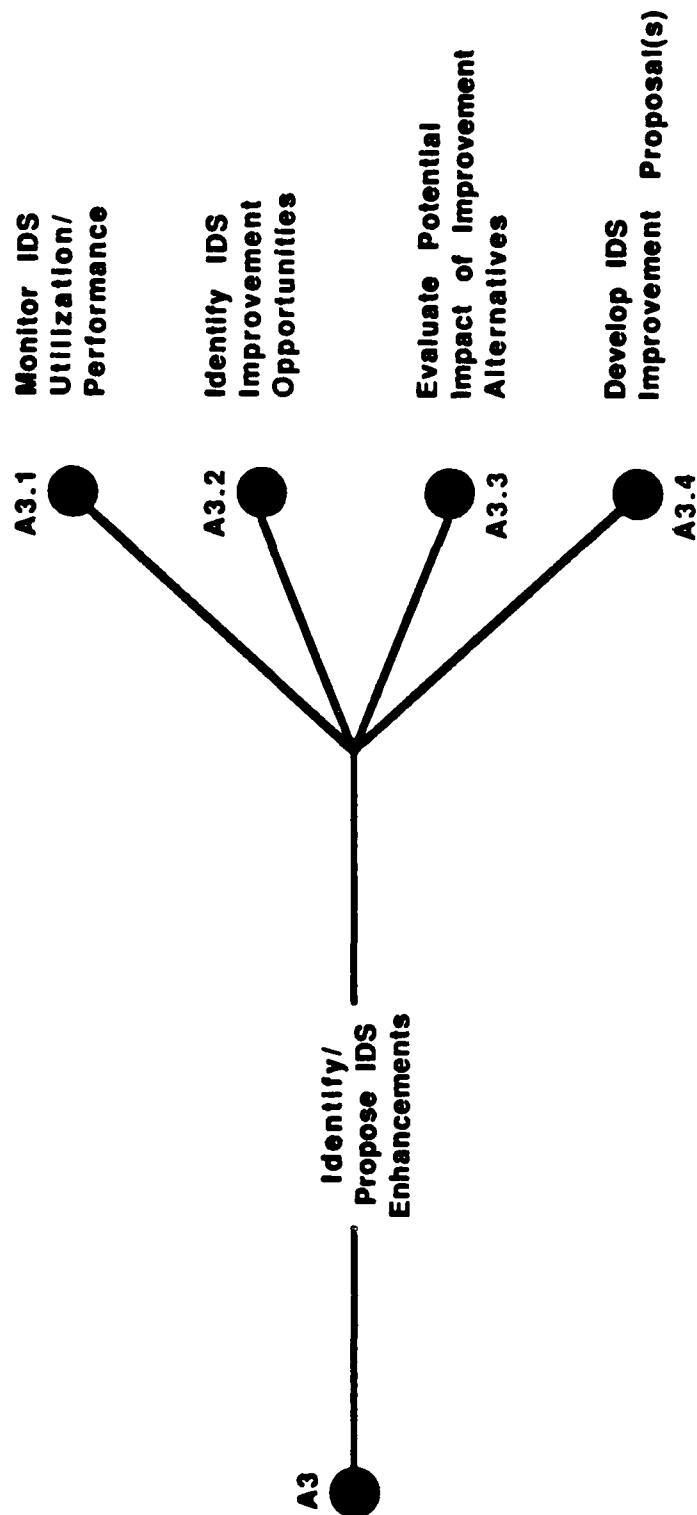
A235.6 Stabilize/Activate New System

A236 DETAIL NODE INDEX

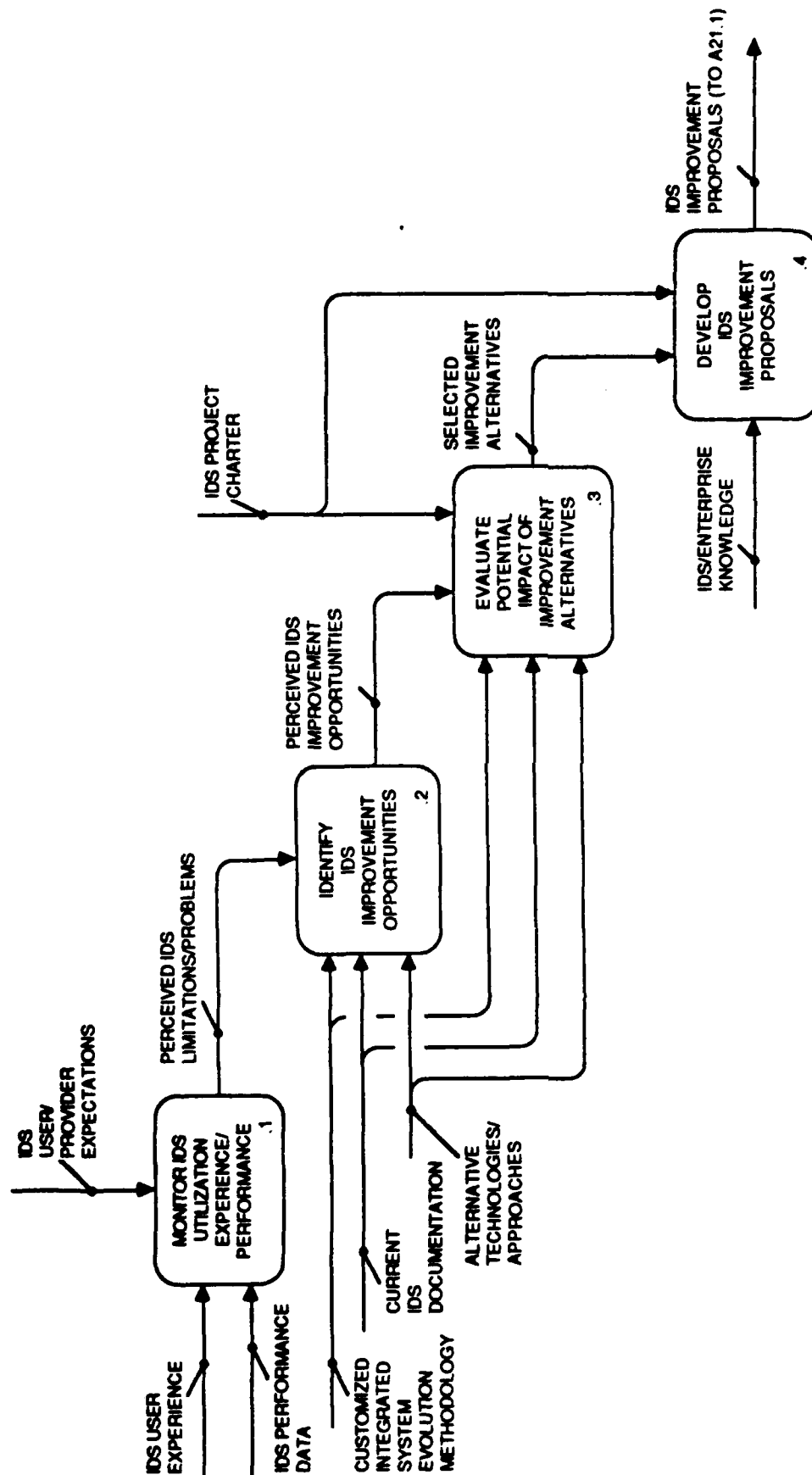
- A236 Optimize Effectiveness/Efficiency of Integrated Information Systems
 - A236.1 Monitor Operational Characteristics of Integrated Information Systems
 - A236.2 Identify Integrated Information Systems' Improvement Opportunities
 - A236.3 Evaluate Alternative Improvement Approaches
 - A236.4 Develop Integrated Information Systems' Improvement Proposal(s)

A3 IDENTIFY/PROPOSE IDS ENHANCEMENTS

As the use of the IDS expands, the IDS itself must be protected from atrophy. New technology will emerge that can potentially improve its functionality. New ways of using it will be discovered with experience. The IDS, along with the Integrated System Evolution Methodology which guides its use, must periodically be updated to take advantage of these advances in technology and understanding.



A3 **NODE TREE (FEO)**



A3 IDENTIFY/PROPOSE IDS ENHANCEMENTS

APPENDIX D

Mechanism/Leaf Node Matrix

ARTIFACT 1
**SDP-0 MECHANISM/
LEAF NODE MATRIX**

**REV. 1
MAY 1987**

ARTIFACT 1 - SDP-0 MECHANISM/LEAF NODE MATRIX

SDP-0 MECHANISMS

SDP-0 LEAF NODES														
	PERFORMANCE ANALYSIS	SMULATION	MODELING	LCA DATA MANAGEMENT	SET ANALYSIS	CODING AND CLASSIFICATION	SURVEY	SYSTEM DESCRIPTION	PROTOTYPING	DATA FLOW ANALYSIS	CONTROL FLOW ANALYSIS	COST-BENEFIT	STRUCTURED DESIGN	DATABASE DESIGN
	TEST	VALIDATION	VERIFICATION	CONSTRUCTION	CONFIGURATION MANAGEMENT	SPECIFICATION LANGUAGE								
A111.1 ANALYZE EXISTING SYSTEMS	●		●											
A111.2 VALIDATE SYMPTOMS AND CONCERNS	●			●										
A111.3 BOUND PROBLEM DOMAIN				●	●									
A111.4 ASSESS TECHNOLOGY				●		●	●							
A111.5 IDENTIFY NEEDS				●				●						
A111.6 EVALUATE NEEDS				●	●									
A121.1 ASSESS TECHNOLOGY OPTIONS				●		●	●							
A121.2 PARTITION PROBLEM DOMAIN			●	●				●						
A121.3 IDENTIFY REQUIREMENTS		●	●	●				●						
A121.4 PERFORM TRIAL AND ERROR EVALUATION OF REQUIREMENTS	●	●		●					●					
A121.5 EVALUATE REQUIREMENTS				●	●									
A211.1 EXAMINE REQUIREMENTS				●										
A211.2 DECOMPOSE SOLUTION			●	●				●		●				
A211.3 ASSESS APPLICABILITY OF TECHNOLOGY		●		●		●	●		●					
A211.4 PERFORM TRADE-OFF ANALYSES OF SOLUTION ALTERNATIVES	●	●		●							●			
A211.5 ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS	●	●		●				●						
A211.6 EVALUATE DESIGN STRUCTURE				●	●									
A221.1 DEFINE INTERNAL INTERFACES				●				●				●		
A221.2 IDENTIFY SUBSYSTEMS AND COMPONENTS				●										
A221.3 ANALYZE SUBSYSTEMS				●				●						
A221.4 PACKAGE COMPONENTS		●		●				●						
A221.5 ESTABLISH COMPONENT SPECIFICATIONS				●				●						
A221.6 EVALUATE COMPONENT SPECIFICATIONS				●	●								●	
A311.1 PREPARE DESIGN FOR IMPLEMENTATION				●									●	
A311.2 PLAN COMPONENT CONSTRUCTION				●									●	
A311.3 FABRICATE PARTS				●										●

D48-01-01

ARTIFACT 1 - SDP-0 MECHANISM/LEAF NODE MATRIX

SDP-0 MECHANISMS

SDP-0 LEAF NODES	PERFORMANCE ANALYSIS	SIMULATION	MODELING	LCA DATA MANAGEMENT	SET ANALYSIS	CODING AND CLASSIFICATION	SURVEY	SYSTEM DESCRIPTION	PROTOTYPING	DATA FLOW ANALYSIS	CONTROL FLOW ANALYSIS	COST-BENEFIT	STRUCTURED DESIGN	DATABASE DESIGN	SYNTHESIS LANGUAGE	CONFIGURATION MANAGEMENT	CONSTRUCTION	VERIFICATION	VALIDATION	TEST
A311.4 ASSEMBLE PARTS																				
A311.5 ASSURE COMPONENT QUALITY																				
A321.1 PREPARE VERIFICATION TEST ENVIRONMENT																				
A321.2 EXERCISE VERIFICATION TEST CASES																				
A321.3 EVALUATE VERIFICATION MEASUREMENTS																				
A321.4 ESTABLISH VERIFICATION CONFORMANCE																				
A331.1 PLAN SYSTEM INTEGRATION																				
A331.2 ASSEMBLE SYSTEM ELEMENTS																				
A331.3 INTEGRATE OTHER SYSTEM ASPECTS																				
A331.4 RECORD AS-BUILT CONFIGURATIONS																				
A331.5 PREPARE SYSTEM ELEMENTS FOR DELIVERY																				
A341.1 PREPARE VALIDATION TEST ENVIRONMENT																				
A341.2 EXERCISE VALIDATION TEST CASES																				
A341.3 EVALUATE VALIDATION MEASUREMENTS																				
A341.4 ESTABLISH VALIDATION CONFORMANCE																				
A411.1 PREPARE ENVIRONMENT																				
A411.2 INSTALL SYSTEM																				
A411.3 TRAIN USERS																				
A411.4 INITIATE MAINTENANCE SUBSYSTEM																				
A411.5 EVALUATE AS-INSTALLED SYSTEM																				
A411.6 OBTAIN USER ACCEPTANCE																				
A421.1 MAKE SYSTEM MODIFICATIONS																				
A421.2 RECORD OPERATIONAL SYSTEM CONFIGURATION																				
A421.3 IDENTIFY SYMPTOMS AND CONCERNS																				
A421.4 ANALYZE PROBLEMS																				

APPENDIX E

SDP-0 Mechanism Analysis

ARTIFACT 2

SDP-0 MECHANISM ANALYSIS

REV. 2
MAY 1987

C87-01-01

MECHANISM TYPE: PERFORMANCE ANALYSIS TOOLS

259

DIAGRAM NODE ID		NAME	USAGE	COMMENTS/QUESTIONS
A11.1		ANALYZE NEEDS	• GATHER DATA ABOUT EXISTING SYSTEMS AND ENVIRONMENTS	
	A11.1.1	ANALYZE EXISTING SYSTEM		
	A11.1.2	VALIDATE SYMPTOMS AND CONCERNS		
A42.1		MAINTAIN SYSTEM	• DIAGNOSING PROBLEMS WITH THE EXISTING SYSTEM • ANALYZING REPORTED OBSERVATIONS ABOUT THE [EXISTING] SYSTEM	
	A42.1.1	MAKE SYSTEM MODIFICATIONS		
	A42.1.3	IDENTIFY SYMPTOMS AND CONCERNS		
<div>PERFORMANCE ANALYSIS TOOLS -- AUTOMATED SUPPORT FOR ANALYSIS OF EXISTING SYSTEM PERFORMANCE, ORIENTED PRIMARILY TOWARD THE REDUCTION AND ANALYSIS OF DATA RATHER THAN THE COLLECTION OF RAW DATA, SINCE EACH SYSTEM WOULD HAVE ITS OWN UNIQUE MECHANISM FOR GATHERING DATA.</div>				

C87-01-02

MECHANISM TYPE: SIMULATION TOOLS

DIAGRAM NODE		NAME	USAGE	COMMENTS/QUESTIONS
ID	ID			
A12.1		DEFINE REQUIREMENTS	<ul style="list-style-type: none">• EVALUATE THE POTENTIAL IMPACT OF REQUIREMENTS• UNCOVER FLAWS IN REQUIREMENTS	
	A12.1.3	IDENTIFY REQUIREMENTS		
	A12.1.4	PERFORM TRIAL AND ERROR EVALUATION OF REQUIREMENTS		
A21.1		ESTABLISH DESIGN STRUCTURE	<ul style="list-style-type: none">• AID THE EVALUATION OF PROPOSED SOLUTIONS AND TECHNOLOGIES	
	A21.1.3	ASSESS APPLICABILITY OF TECHNOLOGY		
	A21.1.4	PERFORM TRADEOFF ANALYSES OF SOLUTION ALTERNATIVES		
A22.1		ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS	<ul style="list-style-type: none">• EVALUATE ALTERNATIVE WAYS OF PACKAGING COMPONENTS	
		DEVELOP SPECIFICATIONS		
	A22.1.4	PACKAGE COMPONENTS		
SIMULATION TOOLS -- TOOLS TO SIMULATE BEHAVIOR OF A SYSTEM AND TO PRODUCE STATISTICS TO SUPPORT THE ANALYSIS OF VARIOUS ASPECTS OF SYSTEM PERFORMANCE, E.G., ICAM DECISION SUPPORT SYSTEM (IDSS).				

MECHANISM TYPE: MODELING TOOLS

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A11.1		ANALYZE NEEDS	<ul style="list-style-type: none"> • TO CREATE REPRESENTATIONS OF THE AS-IS SYSTEM & ITS ENVIRONMENT FOR STUDY & ANALYSIS 	
	A11.1.1	ANALYZE EXISTING SYSTEM		
A12.1		DEFINE REQUIREMENTS	<ul style="list-style-type: none"> • ASSIST IN REPRESENTING BOTH THE AS-IS SYSTEM & ITS ENVIRONMENT AS WELL AS THE RAMIFICATIONS OF POSSIBLE CHANGES TO THE SYSTEM 	
	A12.1.2	PARTITION PROBLEM DOMAIN		
	A12.1.3	IDENTIFY REQUIREMENTS	<ul style="list-style-type: none"> • NOT MENTIONED 	
A21.1		ESTABLISH DESIGN STRUCTURE		
	A21.1.2	DECOMPOSE SOLUTIONS		
MODELING TOOLS -- AUTOMATED SUPPORT FOR FUNCTION, INFORMATION AND PROCESS MODELING, E.G., IDEF012 TECHNIQUES				

**MECHANISM TYPE: LIFE-CYCLE ARTIFACT (LCA)
DATA MANAGEMENT TOOLS (PAGE 1 OF 5)**

262

DIAGRAM NODE ID NAME USAGE COMMENTS/QUESTIONS

A11.1	A11.1.2	ANALYZE NEEDS VALIDATE SYMPTOMS & CONCERNS	<ul style="list-style-type: none">• AID DEVELOPER IN MAINTAINING INFORMATION PRODUCED DURING THE DEVELOPMENT EFFORT: SYMPTOMS, CONCERNS, PROBLEM DOMAINS, TECHNOLOGY OPTIONS, NEEDS, ETC.	
	A11.1.3	BOUND PROBLEM DOMAIN		
	A11.1.4	ASSESS TECHNOLOGY		
	A11.1.5	IDENTIFY NEEDS		
	A11.1.6	EVALUATE NEEDS		
A12.1	A12.1.1	DEFINE REQUIREMENTS ASSESS TECHNOLOGY OPTIONS	<ul style="list-style-type: none">• MAINTAINS INFORMATION SUCH AS TECHNOLOGY OPTIONS, EXTERNAL INTERFACES, REQUIREMENTS, ETC.	
	A12.1.2	PARTITION PROBLEM DOMAIN		
	A12.1.3	IDENTIFY REQUIREMENTS		
	A12.1.4	PERFORM TRIAL & ERROR EVALUATION OF REQUIREMENTS		
	A12.1.5	EVALUATE REQUIREMENTS		
<div>LIFE CYCLE ARTIFACT DATA MANAGEMENT TOOLS -- AUTOMATED SUPPORT FOR POPULATING A LIFE CYCLE ARTIFACT DATABASE, MAINTAINING THE ARTIFACTS (VALIDATION AND INTEGRITY CHECKS) AND MIGRATING THE ARTIFACTS THROUGHOUT THE DEVELOPMENT PROCESS (TRACEABILITY AND CROSS-REFERENCING CHECKS)</div>				

C87-01-05

**MECHANISM TYPE: LIFE-CYCLE ARTIFACT (LCA)
DATA MANAGEMENT TOOLS (PAGE 2 OF 5)**

263

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A21.1		ESTABLISH DESIGN STRUCTURE	<ul style="list-style-type: none"> • TRACK THE REQUIREMENTS DESIGN GOALS, SOLUTION STRUCTURES, TECHNOLOGY DISCIPLINES AND DESIGN STRUCTURE, INCLUDING SYSTEM ELEMENTS AND CAPABILITIES 	
	A21.1.1	EXAMINE REQUIREMENTS		
	A21.1.2	DECOMPOSE SOLUTIONS		
	A21.1.3	ASSESS APPLICABILITY OF TECHNOLOGY		
	A21.1.4	PERFORM TRADE-OFF ANALYSES OF SOLUTION ALTERNATIVES		
	A21.1.5	ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS		
	A21.1.6	EVALUATE DESIGN STRUCTURE		

C87-01-06

**MECHANISM TYPE: LIFE-CYCLE ARTIFACT (LCA)
DATA MANAGEMENT TOOLS (PAGE 3 OF 5)**

DIAGRAM NODE		NAME	USAGE	COMMENTS/QUESTIONS
ID	ID			
A22.1	A22.1.1	DEVELOP SPECIFICATION DEFINE INTERNAL INTERFACES	<ul style="list-style-type: none"> • TRACK THE REQUIREMENTS, DESIGN GOALS, DESIGN STRUCTURE AND COMPONENT SPECIFICATIONS 	
	A22.1.2	IDENTIFY SUBSYSTEMS & COMPONENTS		
	A22.1.3	ANALYZE SUBSYSTEMS		
	A22.1.4	PACKAGE COMPONENTS		
	A22.1.5	ESTABLISH COMPONENT SPECIFICATIONS		
	A22.1.6	EVALUATE COMPONENT SPECIFICATIONS		
A31.1		CONSTRUCT COMPONENTS	<ul style="list-style-type: none"> • TRACK INFORMATION SUCH AS REQUIREMENTS, SPECIFICATION IMPLEMENTABLE DESIGNS, INTERNAL INTERFACES, ETC. 	
	A31.1.1	PREPARE DESIGN FOR IMPLEMENTATION		
	A31.1.2	PLAN COMPONENT CONSTRUCTION		
	A31.1.3	FABRICATE PARTS		
	A31.1.4	ASSEMBLE PARTS		
	A31.1.5	ASSURE COMPONENT QUALITY		

**MECHANISM TYPE: LIFE-CYCLE ARTIFACT (LCA)
DATA MANAGEMENT TOOLS (PAGE 4 OF 5)**

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A32.1		VERIFY SYSTEM ELEMENTS	<ul style="list-style-type: none"> HANDLES INFORMATION SUCH AS INTERFACES, SPECIFICATIONS, PROCEDURES, ACCEPTANCE CRITERIA, ETC. 	
	A32.1.1	PREPARE VERIFICATION TEST ENVIRONMENT		
	A32.1.2	EXERCISE VERIFICATION TEST CASES		
	A32.1.3	EVALUATE VERIFICATION MEASUREMENTS		
A33.1	A32.1.4	ESTABLISH VERIFICATION CONFORMANCE	<ul style="list-style-type: none"> HANDLE INTERFACES, DESIGN STRUCTURES, IMPLEMENTATION DESIGNS, INTEGRATION PLANS, AS-BUILT CONFIGURATIONS, ETC. 	
		INTEGRATE SYSTEM ELEMENTS		
	A33.1.1	PLAN SYSTEM INTEGRATION		
	A33.1.2	ASSEMBLE SYSTEM ELEMENTS		
	A33.1.3	INTEGRATE OTHER SYSTEM ASPECTS		
	A33.1.4	RECORD AS-BUILT CONFIGURATION		

**MECHANISM TYPE: LIFE-CYCLE ARTIFACT (LCA)
DATA MANAGEMENT TOOLS (PAGE 5 OF 5)**

**DIAGRAM NODE
ID**

NAME

USAGE

COMMENTS/QUESTIONS

A34.1		VALIDATE SYSTEM ELEMENTS	<ul style="list-style-type: none"> • HANDLE REQUIREMENTS, TEST PROCEDURES, ETC. 	
	A34.1.1	PREPARE VALIDATION TEST ENVIRONMENT		
	A34.1.2	EXERCISE VALIDATION TEST CASES		
	A34.1.3	EVALUATE VALIDATION MEASUREMENTS		
	A34.1.4	ESTABLISH VALIDATION CONFORMANCE		
A41.1		ESTABLISH SYSTEM	<ul style="list-style-type: none"> • HANDLE EXTERNAL INTERFACES, FIELD TEST PLANS, REQUIREMENTS, ETC. 	
	A41.1.5	EVALUATE AS-INSTALLED SYSTEM		
A42.1		MAINTAIN SYSTEM	<ul style="list-style-type: none"> • HANDLE SYMPTOMS, CONCERNS, PROBLEM DOMAINS, ETC. 	
	A42.1.3	IDENTIFY SYSTEMS & CONCERNS		
	A42.1.4	ANALYZE PROBLEMS		

MECHANISM TYPE: SET ANALYSIS TOOLS

DIAGRAM NODE
ID ID

NAME

USAGE

COMMENTS/QUESTIONS

A11.1	A11.1.3	ANALYZE NEEDS	ANALYZE THE RELATIONSHIPS BETWEEN SYMPTOMS, CONCERNS, PROBLEMS AND PROBLEM DOMAINS	<ul style="list-style-type: none">• ANALYZE THE RELATIONSHIPS BETWEEN SYMPTOMS, CONCERNS, PROBLEMS AND PROBLEM DOMAINS• COMPARE NEEDS AGAINST SYMPTOMS AND CONCERNS• ENSURE COVERAGE AND CONSISTENCY• EVALUATE INTERNAL CONSISTENCY OF REQUIREMENTS• ESTABLISH WHETHER THE REQUIREMENTS ARE CONSISTENT AND COMPLETE RELATIVE TO THE NEEDS• ENSURE THE FINAL DESIGN STRUCTURE IS CONSISTENT WITH THE REQUIREMENTS AND DESIGN GOALS• CHECK FOR CONSISTENCY BETWEEN THE SPECIFICATIONS AND THE REQUIREMENTS AND DESIGN GOALS
	A11.1.6	BOUND PROBLEM DOMAIN EVALUATE NEEDS		
A12.1		DEFINE REQUIREMENTS		
	A12.1.5	EVALUATE REQUIREMENTS		
A21.1		ESTABLISH DESIGN STRUCTURE		<ul style="list-style-type: none">• ENSURE THE FINAL DESIGN STRUCTURE IS CONSISTENT WITH THE REQUIREMENTS AND DESIGN GOALS
	A21.1.6	EVALUATE DESIGN STRUCTURE		
A22.1		DEVELOP SPECIFICATIONS		
	A22.1.6	EVALUATE COMPONENT SPECIFICATIONS		
<div>SET ANALYSIS TOOLS -- CAPABILITIES TO DEFINE SETS OF LIFE CYCLE ARTIFACTS AND TO PERFORM OPERATIONS ON THESE SETS, FOR THE PURPOSE OF PERFORMING SUCH TYPES OF ANALYSES AS CONSISTENCY AND COMPLETENESS CHECKING OF THE ARTIFACTS.</div>				

MECHANISM TYPE: SET ANALYSIS TOOLS (cont'd.)

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A42.1	A42.1.4	MAINTAIN SYSTEM ANALYZE PROBLEMS	<ul style="list-style-type: none"> • ESTABLISHING CLUSTERS OF SYMPTOMS AND CONCERNS RELATIVE TO SOME PROBLEM DOMAIN 	

MECHANISM TYPE: CODING AND CLASSIFICATION TOOLS

269

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A11.1		ANALYZE NEEDS	<ul style="list-style-type: none"> • ORGANIZES INFORMATION THAT SURVEY TOOLS COLLECT 	
A12.1	A11.1.4	ASSESS TECHNOLOGY		
		DEFINE REQUIREMENTS	<ul style="list-style-type: none"> • ASSIST IN TECHNOLOGY ASSESSMENT ACTIVITIES 	
	A12.1.1	ASSESS TECHNOLOGY OPTIONS	<ul style="list-style-type: none"> • WHAT TECHNOLOGY OPTIONS ARE STILL APPLICABLE TO THE PROBLEM BEING ADDRESSED 	
A21.1		ESTABLISH DESIGN STRUCTURE	<ul style="list-style-type: none"> • (NO FURTHER DESCRIPTION) 	
	A21.1.3	ASSESS APPLICABILITY OF TECHNOLOGY		
CODING AND CLASSIFICATION TOOLS -- AUTOMATED FOR CODING AND CLASSIFYING INFORMATION TO BE USED IN SELECTING TECHNOLOGY FOR DEVELOPMENT OF A SYSTEM ELEMENT, E.G., TO A LIMITED EXTENT, GROUP TECHNOLOGY SUPPORT SYSTEM (GTSS).				

C87-01-12

MECHANISM TYPE: SURVEY TOOLS

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A11.1		ANALYZE NEEDS	<ul style="list-style-type: none"> • GATHER INFORMATION ABOUT TECHNOLOGY THAT MAY BE APPLICABLE TO THE PROJECT 	
	A11.1.4	ASSESS TECHNOLOGY		
A12.1		DEFINE REQUIREMENTS	<ul style="list-style-type: none"> • ASSIST IN TECHNOLOGY ASSESSMENT ACTIVITIES 	
	A12.1.1	ASSESS TECHNOLOGY OPTIONS		
A21.1		ESTABLISH DESIGN STRUCTURE	<ul style="list-style-type: none"> • [TECHNOLOGY ASSESSMENT TOOLS ARE] USED TO EVALUATE THE APPLICABILITY OF TECHNOLOGY DISCIPLINES 	
	A21.1.3	ASSESS APPLICABILITY OF TECHNOLOGY		

SURVEY TOOLS -- MECHANISMS TO SUPPORT THE SEARCHING OF DATABASES FOR THE AVAILABILITY OF SPECIFIC TECHNOLOGIES, DOCUMENTS AND DESCRIPTIONS OF SYSTEMS, I.E., ARPANET

TECHNOLOGY ASSESSMENT TOOLS -- THE CLASS OF TOOLS USED TO ASSIST IN THE EVALUATION AND SELECTION OF TECHNOLOGY FOR USE IN SYSTEM DEVELOPMENT.

MECHANISM TYPE: SYSTEM DESCRIPTION TOOLS

271

DIAGRAM NODE
ID ID

NAME

USAGE

COMMENTS/QUESTIONS

A11.1		ANALYZE NEEDS	• TO DEFINE THE NEEDS THAT HAVE BEEN IDENTIFIED	
	A11.1.5	IDENTIFY NEEDS		
A12.1		DEFINE REQUIREMENTS	• TO DEFINE EXTERNAL INTERFACES AND REQUIREMENTS	
	A12.1.2	PARTITION PROBLEM DOMAIN		
	A12.1.3	IDENTIFY REQUIREMENTS		
A21.1		ESTABLISH DESIGN STRUCTURE	• NOT MENTIONED	
	A21.1.2	DECOMPOSE SOLUTIONS		
	A21.1.5	ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS		
SYSTEM DESCRIPTION TOOLS -- TOOLS TO SUPPORT THE DEFINITION OF SYSTEMS AND THEIR STRUCTURE; USUALLY PROVIDE A SPECIAL LANGUAGE OR LANGUAGES, E.G., EXTENDED REQUIREMENTS EVALUATION AND VALIDATION SYSTEM (EREVS) AND CADSAT.				

C87-01-14

MECHANISM TYPE: SYSTEM DESCRIPTION TOOLS (cont'd.)

272

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A22.1		DEVELOP SPECIFICATIONS	• NOT MENTIONED	
	A22.1.1	DEFINE INTERNAL INTERFACE		
	A22.1.3	ANALYZE SUBSYSTEMS		
	A22.1.4	PACKAGE COMPONENTS		
	A22.1.5	ESTABLISH COMPONENT SPECIFICATIONS		
A33.1		INTEGRATE SYSTEM ELEMENTS	• USEFUL IN DOCUMENTING THE AS-BUILT CONFIGURATION	
	A33.1.4	RECORD AS-BUILT CONFIGURATION		

MECHANISM TYPE: PROTOTYPING TOOLS

273

DIAGRAM NODE ID NAME USAGE COMMENTS/QUESTIONS

A12.1	<p>DEFINE REQUIREMENTS</p> <p>PERFORM TRIAL & ERROR EVALUATION OF REQUIREMENTS</p>	<ul style="list-style-type: none"> • TO BUILD REQUIREMENTS PROTOTYPES, WHICH ARE RAPIDLY DEVELOPED MINI-SYSTEMS THAT IMPLEMENT CERTAIN IMPORTANT ASPECTS OF THE SYSTEM UNDER DEVELOPMENT. 	
A21.1	<p>ESTABLISH DESIGN STRUCTURE</p> <p>ASSESS APPLICABILITY OF TECHNOLOGY</p> <p>ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS</p>	<ul style="list-style-type: none"> • TO AID THE EVALUATION OF PROPOSED SOLUTIONS AND TECHNOLOGIES 	
			<p>PROTOTYPING TOOLS -- SUPPORT FOR DEVELOPING PROTOTYPES DURING REQUIREMENTS DEFINITION AND PRELIMINARY DESIGN.</p>

C87-01-16

MECHANISM TYPE: DATA FLOW ANALYSIS TOOLS

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A21.1	A21.1.2	ESTABLISH DESIGN STRUCTURE DECOMPOSE SOLUTIONS	• AID THE EVALUATION OF PROPOSED SOLUTIONS AND TECHNOLOGIES	DATA FLOW ANALYSIS TOOLS -- AUTOMATED SUPPORT FOR ANALYZING THE FLOW OF DATA ITEMS WITHIN A SYSTEM, E.G., COMPUTER-AIDED DESIGN AND SYSTEM ANALYSIS TOOL (CADSAT).

MECHANISM TYPE: CONTROL FLOW ANALYSIS TOOLS

275

DIAGRAM NODE		NAME	USAGE	COMMENTS/QUESTIONS
ID	ID			
A21.1		ESTABLISH DESIGN STRUCTURE	• AID THE DECOMPOSITION OF SOLUTION STRUCTURES	
	A21.1.2	DECOMPOSE SOLUTIONS		
CONTROL FLOW ANALYSIS TOOLS -- AUTOMATED SUPPORT FOR ANALYZING THE ORDER IN WHICH SYSTEM OPERATIONS ARE PERFORMED, E.G., SYSTEM'S ARCHITECT APPRENTICE (SARA).				

C87-01-18

MECHANISM TYPE: COST/BENEFIT ANALYSIS TOOLS

276

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A21.1	A21.1.4	ESTABLISH DESIGN STRUCTURE PERFORM TRADE-OFF ANALYSIS OF SOLUTION ALTERNATIVES	• NOT MENTIONED	
				COST/BENEFIT ANALYSIS TOOLS -- ASSISTANCE IN ANALYZING THE COSTS, IN TERMS OF RESOURCES, TRADE-OFFS, ETC., OF ALTERNATE DESIGN CHOICES AGAINST THE BENEFITS TO BE DERIVED FROM THEM.

C87-01-19

MECHANISM TYPE: STRUCTURED DESIGN TOOLS

277

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A21.1		ESTABLISH DESIGN STRUCTURE	• DEVELOPING THE DESIGN STRUCTURE	
A22.1	A21.1.5	ALLOCATE TECHNOLOGY TO SYSTEM ELEMENTS	• DESIGN INTERNAL INTERFACES	
	A22.1.1	DEVELOP SPECIFICATIONS DEFINE INTERNAL INTERFACES		
A31.1		CONSTRUCT COMPONENTS PREPARE DESIGN FOR IMPLEMENTATION	• PRODUCING IMPLEMENTABLE DESIGNS	
A33.1	A31.1.1	INTEGRATE SYSTEM ELEMENTS RECORD "AS-BUILT" CONFIGURATIONS	• DOCUMENTING THE "AS-BUILT" CONFIGURATION	
	A33.1.4			
STRUCTURED DESIGN TOOLS -- AUTOMATED SUPPORT FOR THE STRUCTURED DESIGN METHODOLOGY, E.G., AUTOMATED INTERACTIVE DESIGN AND EVALUATION SYSTEM (AIDES).				

C87-01-20

MECHANISM TYPE: DATABASE DESIGN TOOLS

278

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A22.1		DEVELOP SPECIFICATIONS	• DEVELOP SCHEMA SPECIFICATION	
	A22.1.5	ESTABLISH COMPONENT SPECIFICATIONS	• SUPPORT PROGRAM SPECIFICATIONS	
A31.1		CONSTRUCT COMPONENTS	• DATABASE DESIGN TOOLS DEVELOP INTERNAL DATABASE SCHEMA	
	A31.1.1	PREPARE DESIGN FOR IMPLEMENTATION		
<div> <p>DATABASE DESIGN TOOLS -- SUPPORT FOR THE DESIGN OF DATABASE SCHEMA AND AUTOMATED GENERATION OF DATA DESCRIPTION LANGUAGE (DDL) STATEMENTS, E.G., AN AUTOMATED VERSION OF THE IDEF-1 TECHNIQUE.</p> </div>				

C87-01:21

MECHANISM TYPE: SPECIFICATION LANGUAGES

COMMENTS/QUESTIONS

USAGE

NAME

DIAGRAM NODE
ID ID

- TO REPRESENT SPECIFICATIONS IN MACHINE PROCESSIBLE FORM

- FOR DESCRIBING DETAILED CHARACTERISTICS OF THE PART

DEVELOP
SPECIFICATIONSESTABLISH COMPONENT
SPECIFICATIONS

A22.1.5

CONSTRUCT
COMPONENTSPREPARE DESIGN FOR
IMPLEMENTATION

A31.1.1

SPECIFICATION LANGUAGES -- MACHINE PROCESSIBLE LANGUAGES
USED TO DESCRIBE COMPONENT SPECIFICATIONS, E.G., SOFTWARE
DESIGN AND DOCUMENTATION LANGUAGE (SDDL).

C87-01-22

MECHANISM TYPE: CONFIGURATION MANAGEMENT TOOLS

280

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A22.1		DEVELOP SPECIFICATIONS		
	A22.1.6	EVALUATE COMPONENT SPECIFICATIONS		• NOT MENTIONED
A31.1		CONSTRUCT COMPONENTS		• NOT MENTIONED
	A31.1.5	ASSURE COMPONENT QUALITY		
A32.1		VERIFY SYSTEM ELEMENTS		• FOR TRACKING THE STATUS OF SYSTEM ELEMENTS UNDERGOING VERIFICATION TESTING
	A32.1.4	ESTABLISH VERIFICATION CONFORMANCE		
A33.1		INTEGRATE SYSTEM ELEMENTS		• USED TO CONTROL THE RESULTING CONFIGURATION
	A33.1.4	RECORD AS-BUILT CONFIGURATIONS		
CONFIGURATION MANAGEMENT TOOLS -- AUTOMATED CAPABILITY FOR MANAGING CONFIGURATIONS OF THE SYSTEM DESIGN AND CONSTRUCTED ELEMENTS, E.G., CHANGE CONTROL AND REPORTING SYSTEM (CCARS).				

C87-01-23

MECHANISM TYPE: CONFIGURATION MANAGEMENT TOOLS (cont'd.)

281

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A34.1		VALIDATE SYSTEM ELEMENTS	• USED TO TRACK THE STATUS OF THE SYSTEM ELEMENT UNDERGOING VALIDATION	
A34.1.4		ESTABLISH VALIDATION CONFORMANCE		
A41.1		ESTABLISH SYSTEM	• TO TRACK THE STATUS OF THE SYSTEM DURING FINAL TESTING	
A41.1.5		EVALUATE AS- INSTALLED SYSTEM		
A42.1		MAINTAIN SYSTEM	• TO CONTROL CHANGES TO THE OPERATIONAL SYSTEM	
	A42.1.2	RECORD OPERATIONAL SYSTEMS CONFIGURATION		

C87-01-24

MECHANISM TYPE: CONSTRUCTION TOOLS

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A31.1		CONSTRUCT COMPONENTS	<ul style="list-style-type: none"> • USED TO BUILD, ASSEMBLE AND INSPECT THE PARTS • ASSEMBLE THE ELEMENTS • INTEGRATE OPERATIONAL SUPPORT ITEMS 	
	A31.1.3	FABRICATE PARTS		
	A31.1.4	ASSEMBLE PARTS		
	A31.1.5	ASSURE COMPONENT QUALITY		
A33.1		INTEGRATE SYSTEM ELEMENTS		
	A33.1.2	ASSEMBLE SYSTEM ELEMENTS	<div>CONSTRUCTION TOOLS -- TOOLS USED TO BUILD SYSTEM ELEMENTS, E.G., IN SOFTWARE DEVELOPMENT, EDITORS, COMPILERS AND DEBUGGERS.</div>	
	A33.1.3	INTEGRATE OTHER SYSTEM ASPECTS		

AD-A195 851

KNOWLEDGE-BASED INTEGRATED INFORMATION SYSTEMS
DEVELOPMENT METHODOLOGIES. (U) MASSACHUSETTS INST OF
TECH CAMBRIDGE A GUPTA ET AL. DEC 87 MIT-KBIISE-2

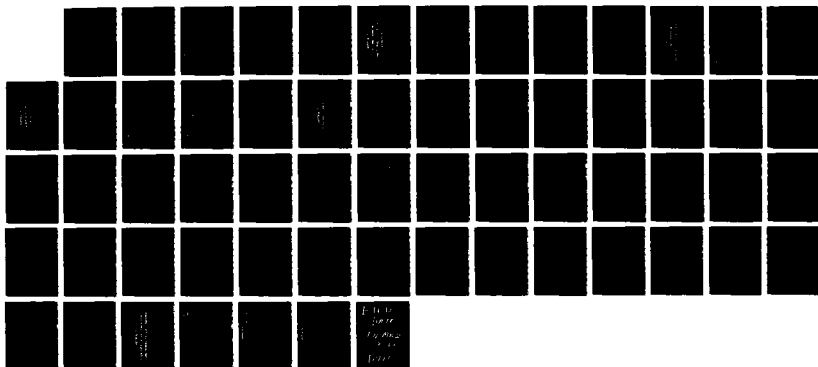
4/4

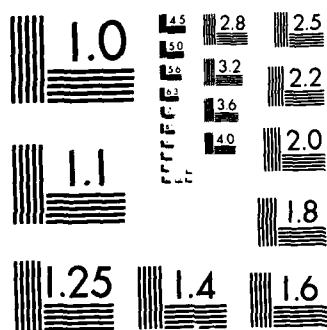
UNCLASSIFIED

DTR557-85-C-00083

F/G 12/7

ML





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

MECHANISM TYPE: VERIFICATION TOOLS

283

DIAGRAM ID	NODE ID	NAME	USAGE	COMMENTS/QUESTIONS
A32.1		VERIFY SYSTEM ELEMENTS PREPARE VERIFICATION TEST ENVIRONMENT EXERCISE VERIFICATION TEST CASES	<ul style="list-style-type: none"> EITHER SPECIALLY DEVELOPED TOOLS FOR THE SYSTEM ELEMENT UNDER TEST OR MORE GENERAL TEST TOOLS AVAILABLE IN THE ENGINEERING ENVIRONMENT 	
			VERIFICATION TOOLS -- NO DEFINITION IN SDP-0 GLOSSARY	

C87-01-26

MECHANISM TYPE: VALIDATION TOOLS

284

DIAGRAM NODE ID		NAME	USAGE	COMMENTS/QUESTIONS
A34.1		VALIDATE SYSTEM ELEMENTS	<ul style="list-style-type: none">• PURPOSE: DOES SYSTEM SATISFY ITS REQUIREMENTS• ENCOMPASSES: TEST PROCEDURES, TEST FIXTURES, TEST CASES & ACCEPTANCE CRITERIA• SPECIAL PURPOSE TOOLS & GENERAL ENVIRONMENT TOOLS	
A34.1.1		PREPARE VALIDATION TEST ENVIRONMENT		
A34.1.2		EXERCISE VALIDATION TEST CASES		
VALIDATION TOOLS -- AUTOMATED SUPPORT FOR TESTS USED TO VALIDATE A SYSTEM ELEMENT AGAINST ITS REQUIREMENTS.				

CB7-01-27

MECHANISM TYPE: TEST TOOLS

285

DIAGRAM NODE ID		NAME	USAGE	COMMENTS/QUESTIONS
A41.1		ESTABLISH SYSTEM	• TO EVALUATE THE AS-INSTALLED SYSTEM, INCLUDING ACCEPTANCE AND FIELD TESTING	
A41.1.5		EVALUATE AS- INSTALLED SYSTEM		
TEST TOOLS -- THE CLASS OF TOOLS USED TO ASSIST IN THE VERIFICATION, VALIDATION AND FIELD TESTING OF SYSTEM ELEMENTS.				

C87-01-28

APPENDIX F

SDP-0 Mechanism Functionality Classes

ARTIFACT 3

SDP-0 MECHANISM

FUNCTIONALITY CLASSES/ DEFINITIONS

**REV. 3
MAY 1987**

ARTIFACT 3 - SDP-0 MECHANISM FUNCTIONALITY CLASSES/DEFINITIONS**FUNCTIONALITY CLASS****DESCRIPTION/DEFINITION****FUNCTION MODELING**

A method/tool for developing structured representations of the processes performed by a system (Business Area or Computer Application). Function Models may be developed using hierarchical decomposition and object flow diagramming techniques which, in addition to representing processes, indicate the flow of objects and/or information about those objects among the processes. Example: IDEF-0 Based Tools.

INFORMATION MODELING

A method/tool for developing structured representations of the kinds of information required by a system and the interrelationships among the kinds of information. An Information Model generally identifies entity classes, attribute classes and relation classes in a normalized form. Example: IDEF-1 Based Tools.

DYNAMICS MODELING

A method/tool for developing structured representations of the time sensitive relationships between processes (events) such as the activation/deactivation sequence, triggers and constraints, activation/deactivation thresholds, volume and rates of object flow, resource consumption per activation cycle and conditional paths. Example: IDEF-2 Based Tools.

STATE TRANSITION MODELING

A method/tool for developing structured representations which show, for any object, the allowable states of the object, allowable changes in states and constraints governing state changes (such as the allowable order of change) and reentry to previous states.

SPECIFICATION REPRESENTATION

A method/tool which supports specification of a design used for construction of a system and/or system components.
Example: SDDL (Software Design and Documentation Language).

ARTIFACT 3 - SDP-0 MECHANISM FUNCTIONALITY CLASSES/DEFINITIONS

<u>FUNCTIONALITY CLASS</u>	<u>DESCRIPTION/DEFINITION</u>
SIMULATION ANALYSIS	A method/tool used to construct a mathematical profile of the projected behavior of system components based upon parameters provided through a Dynamics Model and/or State Transition Model. Example: IDSS 2.0 (Integrated Decision Support System)
CODING AND CLASSIFICATION	A method/tool which assists in the coding and classification of system components based upon their respective characteristics. Example: GTSS (Group Technology Support System)
CONFIGURATION CONTROL	A method/tool which controls the release and usability of system components and their various versions. Example: CCARS (Change Control and Reporting System)/SCCS (Source Code Control System)
PERFORMANCE ANALYSIS	A method/tool for monitoring and tracking the performance of implemented system components.
LIFE CYCLE ARTIFACT MANAGEMENT	A method/tool for capturing, retaining and facilitating the use/maintenance of project/system documentation throughout the system life cycle. Also employed to assure completeness and consistency of life cycle artifact content.
DATABASE GENERATION	A method/tool which translates an Information Model into a physical database design suitable to the particular Database Management System(s) used by a system. Example: DEFINE.IT
SOFTWARE GENERATION	A method/tool which generates executable code based upon program specifications. Example: TRANSFORM.

ARTIFACT 3 - SDP-0 MECHANISM FUNCTIONALITY CLASSES/DEFINITIONS

<u>FUNCTIONALITY CLASS</u>	<u>DESCRIPTION/DEFINITION</u>
ECONOMETRIC ANALYSIS	<p>A method/tool which facilitates an analysis of the economic impact of a system on its environment. It assists in analyzing costs (in terms of resources, trade-offs, etc.) of alternate design choices against the benefits to be derived from them.</p> <p>Example: CDEF (Cost Definition System)</p>
CLUSTER ANALYSIS	<p>A method/tool which uses algorithmic processing for grouping system components into clusters/classes based upon similarity and/or commonality of their respective characteristics.</p>
SURVEY SUPPORT	<p>A method/tool which aids in the construction and execution of surveys/interviews and capturing of the results.</p>
TEST CASE GENERATION	<p>A method/tool which generates test plans, test data and expected results through the interpretation of construction specifications and/or constructed software components.</p>

APPENDIX G

SDP-0 Mechanism Functionality Class Matrix

ARTIFACT 4
**SDP-0 MECHANISM/
FUNCTIONALITY CLASS MATRIX**

**REV. 1
MAY 1987**

ARTIFACT 4 SDP-O MECHANISM/ FUNCTIONALITY CLASS MATRIX

SDP-O MECHANISMS

FUNCTIONALITY CLASSES

FUNCTION MODELING	INFORMATION MODELING	DYNAMICS MODELING	STATE TRANSITION MODELING	SPECIFICATION GENERATION	SIMULATION ANALYSIS	CODING & CLASSIFICATION	CONFIG. MANAGEMENT	PERFORM. ANALYSIS	LC ARTIFACT MANAGEMENT	DATABASE GENERATION	SOFTWARE GENERATION	ECONOMETRIC ANALYSIS	CLUSTER ANALYSIS	SURVEY SUPPORT	TEST CASE GENERATION
PERFORMANCE ANALYSIS								●							
SIMULATION					●										
MODELING	●	●	●												
LIFE-CYCLE ARTIFACT (LCA) DATA MANAGEMENT									●						
SET ANALYSIS													●		
CODING AND CLASSIFICATION						●									
SURVEY														●	
SYSTEM DESCRIPTION	●	●	●	●	●	●		●	●	●			●	●	●
PROTOTYPING								●	●	●	●				●
DATA FLOW ANALYSIS	●														
CONTROL FLOW		●													
COST-BENEFIT												●			
STRUCTURED DESIGN	●	●								●					
DATABASE DESIGN		●	●							●					
SPECIFICATION LANGUAGE				●											
CONFIGURATION MANAGEMENT							●								
CONSTRUCTION										●	●				
VERIFICATION					●									●	●
VALIDATION								●	●	●	●				
TEST					●										●

APPENDIX H

Candidate Tool Survey List

ARTIFACT 5
CANDIDATE TOOL
SURVEY LIST

REV. 4
MAY 1987

ARTIFACT 5 - CANDIDATE TOOL SURVEY LIST

296

REF. #	TOOL NAME	MANUFACTURER/CONTACT
1	INFORMATION PLANNER	KnowledgeWare, Inc. (404) 231-8575
2	ISS-THREE	UCCEL Contact: Greg Strouse (214) 353-7457
3	INFORMATION ENGINEERING WORKBENCH	KnowledgeWare, Inc. (404) 231-8575
4	EXCELLERATOR	Index Technology Corporation (617) 491-2100
5	DESIGN AID	Nastec Corporation Contact: Shawn McLaughlin (800) 872-8296
6	AUTO-MATE PLUS	Learmonth & Burchett Management Systems, Inc. Contact: Darby Terry (800) 231-7515
7	LEVERAGE	D. Appleton Company, Inc. Contact: Dave Schoeff (213) 546-7575
8	PRECISE/IE	CDC Contact: James Anderson (612) 853-8772
9	STRADIS/DRAW	McDonnell Douglas Information Systems Group Contact: Wanita Auto (800) 325-1087
10	SYSTEM DEVELOPER'S PRO KIT	McDonnell Douglas Information Systems Group Contact: Wanita Auto (800) 325-1087
11	DATA DESIGNER II	KnowledgeWare, Inc. (404) 231-8575
12	CDEF	Price Waterhouse Contact: Karen Goodfriend (213) 236-3157
13	GAMMA	KnowledgeWare, Inc. (404) 231-8575
14	TELON	Pansophic Systems, Inc. Contact: Al Martelo (213) 640-2910
15	TRANSFORM	Transform Logic Corporation Contact: D. Kirk Boudreaux (602) 948-2600
16	STRATEGIC SYSTEMS PLANNING	Holland Systems Corporation Contact: Kerry Baxter (313) 995-9595
17	LOGICAL DATABASE DESIGN	Holland Systems Corporation Contact: Kerry Baxter (313) 995-9595

D55-01-01

ARTIFACT 5 - CANDIDATE TOOL SURVEY LIST

REF. #	TOOL NAME	MANUFACTURER/CONTACT
18	PACBASE	CGI Systems, Inc. Contact: James Osborn (415) 954-8544
19	APPLICATION PRODUCTIVITY SYSTEMS	Sage Software, Inc. Contact: Harold Daniels (800) 638-8703
20	MAZDAMON	UOCEL Contact: Greg Strouse (214) 353-7457
21	DEFINE.IT	Solion, Inc. Contact: Pacific Information Management, Inc. (213) 829-3380
22	INFORMATION MANAGEMENT FACILITY	Boole & Babbage Contact: Carol Kaplan (408) 735-9550
23	XPF/COBOL	Boole & Babbage Contact: Carol Kaplan (408) 735-9550
24	TUTSIM	Applied i (415) 325-4800
25	PAC II	AGS Management Systems Contact: Judy Armstrong (714) 476-2464
26	BLUE/60 THE DATA MODELER	Abvent (213) 659-5157
27	LOGISCOPE	Verilog Contact: Lon Lawson (703) 354-0371
28	ASA	Verilog Contact: Lon Lawson (703) 354-0371
29	TAGS	Teledyne Brown Engineering Contact: Neil Owen (800) 633-4675
30	STATEMATE 1	ADCAD (617) 576-5732
31	COSTAR	Softstar Systems Contact: Dan Ligett (603) 672-0987
32	REFINE	Reasoning Systems Contact: Julianne Ekstrom (415) 494-6201
33	KEYONE	LPS +39.11 831830/885934 TORINO, ITALY
34	ISTAR	Imperial Software Technology, Ltd. (414) 5818 155 LONDON, ENGLAND

ARTIFACT 5 - CANDIDATE TOOL SURVEY LIST

298

REF. #	TOOL NAME	MANUFACTURER/CONTACT
35	MACH I	Tominy, Inc. Contact: Sandra Moonert (513) 984-6605
36	PROMOD	Promod Contact: Thomas Scott (714) 855-3046
37	AMPLIFY CONTROL	CaseWare Contact: Ron Kester (714) 754-0308
38	SOFTWARE THROUGH PICTURES	Interactive Development Environment Contact: Mike Stoeckig (415) 543-0900
39	INFORMATION ENGINEERING FACILITY (IEF)	Texas Instruments Contact: John Buselli (214) 575-4407
40	SLAM II	Pritsker & Associates, Inc. Contact: Kay A. Smith (317) 463-5557
41	DESIGN/I	Arthur Anderson & Co.
42	SIMPLAN	Simplan Systems, Inc. Contact: John Crane (800) 334-8660
43	DATA BASE PLUS	Tominy, Inc. Contact: Sandra Moonert (513) 984-6605
44	MANAGER FAMILY	Manager Software Products, Inc. Contact: James Walsh (617) 863-5800
45	EIFFEL	Interactive Software Engineering, Inc. Contact: Bertrand Mayer (805) 685-1006
46	E R PACKAGE	Chen and Associates (504) 928-5765

D55-01-03

APPENDIX I

Tool Fact Sheets

ARTIFACT 6

TOOL FACT SHEETS

REV. 2
MAY 1987

TOOL FACT SHEET

REF. #: 1	TOOL NAME: INFORMATION PLANNER
SOURCE: KNOWLEDGEWARE, INC.	
FIRST RELEASE DATE: DECEMBER 1984	CURRENT RELEASE LEVEL: 2.2
APPROX. INSTALLATIONS: 125	PRICE RANGE: \$35,000 LESS DISCOUNTS

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	MVS			
IBM	VM/CMS			
DEC/VAX				

DESCRIPTION

INFORMATION PLANNER helps develop an information strategy for the enterprise. Using an enterprise model and the analysis capabilities of INFORMATION PLANNER, an Information Strategy can be developed from a top-down perspective. INFORMATION PLANNER can be used with various information planning methodologies such as IBM's Business Systems Planning (BSP) and James Martin's Strategic Data Planning Methodology.

The analysis technique in INFORMATION PLANNER identifies common information needs among enterprise processes. Using information about enterprise goals stored in its knowledge base, INFORMATION PLANNER prioritizes information projects that implement shared databases based on the degree to which each project addresses the enterprise goals and critical success factors.

INFORMATION PLANNER supports function modeling and high level information modeling, and has five model analysis techniques that analyze planning data. The first two, Exception Analysis and Level Consistency Analysis, are model validation techniques to help ensure that the enterprise model provides an accurate, consistent and complete foundation for decision making. The other three, Affinity Analysis, Project Action Analysis and Project Ranking Analysis, are techniques that provide decision support and aid in the development of an Information Strategy.

TOOL FACT SHEET**REF. #:2** **TOOL NAME: ISS-THREE****SOURCE: UCCEL****FIRST RELEASE DATE: MAY 1986****CURRENT RELEASE LEVEL: 1.2****APPROX. INSTALLATIONS: 30****PRICE RANGE: \$35,000 LESS DISCOUNTS****HOST ENVIRONMENT(S)**

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	MVS/SP MVS/XA			
IBM PC/XT, PC/AT AND PC/3270	DOS 2.0			

DESCRIPTION

ISS-THREE is a capacity management tool that integrates the MVS Analyzer and Capacity Planner. The MVS Analyzer, a mainframe tool, extracts, summarizes and reports on system activity for MVS/XA and MVS/SP operating systems.

The Capacity Planner is a PC based tool designed to model any computer system. It employs expert systems techniques to recommend appropriate equipment configurations to meet specified objectives. The system permits "what if" analysis to compare alternative scenarios.

TOOL FACT SHEET

303

REF. #: 3 **TOOL NAME:** INFORMATION ENGINEERING WORKBENCH
SOURCE: KNOWLEDGEWARE, INC.
FIRST RELEASE DATE: MAY 1986 **CURRENT RELEASE LEVEL:** 3.0
APPROX. INSTALLATIONS: 1700 **PRICE RANGE:** \$7500 LESS DISCOUNTS

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/AT AND COMPATIBLES	MS/DOS			ASCII IMPORT/EXPORT

DESCRIPTION

Based on James Martin's Information Engineering approach, IEW is a structured analysis tool that supports Function Modeling, Data Modeling and Function/Data Dynamics Modeling. Function Modeling graphics are compatible with DeMarco/Yourdon Data Flow Diagrams (DFD's). Data Modeling graphics are a variation of Chen's Entity Relationship (ER) notation augmented with Martin's P-Frames. The user is provided with a selectable ICON set. Function/Data Dynamics are represented via Martin's Action Diagrammer notation. Graphics are generated from stored model descriptions, i.e., a stored fact base. Rule enforcement via Prolog inference tables enforces consistency among levels within a model, and among related models.

TOOL FACT SHEET

304

REF. #: 4 **TOOL NAME:** EXCELLERATOR
SOURCE: INDEX TECHNOLOGY CORPORATION
FIRST RELEASE DATE: AUGUST 1984 **CURRENT RELEASE LEVEL:** 1.7
APPROX. INSTALLATIONS: 5,000 **PRICE RANGE:** \$8,400 - \$12,500

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/XT, IBM PC/AT AND COMPATIBLES	MS/DOS			

DESCRIPTION

EXCELLERATOR provides integrated graphics for function and data modeling, reporting, documentation and screen/report design facilities for the analysis and design of systems. It supports four new graphics developed for real time system design, transformation graphics, state transition diagrams and block diagrams. It also supports two methodologies for real time design (Ward & Millor and Hatley).

EXCELLERATOR provides a tool for creating Data Flow Diagrams, Structure Charts, Structure Diagrams, Logical Data Models, Entity Relationships, Data Models and Presentation Graphics.

D17-01-04

TOOL FACT SHEET**REF. #:**5 **TOOL NAME:** DESIGN AID**SOURCE:** NASTEC CORPORATION**FIRST RELEASE DATE:** 1979**CURRENT RELEASE LEVEL:** 3.55**APPROX. INSTALLATIONS:** 1000**PRICE RANGE:** \$6,900 LESS DISCOUNTS**HOST ENVIRONMENT(S)**

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/XT, PC/AT, PC/3270 AND COMPATIBLES				

DESCRIPTION

DESIGN AID is an automated design and documentation tool which is used by systems and programmer analysts. DESIGN AID's dictionary stores and reports on all project definitions developed during the systems design process. DESIGN AID's integrated graphics and text editor, as well as its validation and analysis capabilities are fully integrated into the system. DESIGN AID is the foundation tool in Nastec's CASE 2000 product line. DESIGN AID supports analysis and design techniques with facilities for Yourdon Structured Analysis and Design and Orr Data Structured Systems Development.

TOOL FACT SHEET

306

REF. #: 6 **TOOL NAME:** AUTO-MATE PLUS
SOURCE: LEARMONTH & BURCHETT MANAGEMENT SYSTEMS, INC.
FIRST RELEASE DATE: SEPTEMBER 1985 **CURRENT RELEASE LEVEL:**
APPROX. INSTALLATIONS: **PRICE RANGE:** \$8,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/AT AND COMPATIBLES	MS/DOS 3.2			

DESCRIPTION

AUTO-MATE PLUS is an integrated software product for supporting system analysis, logical design, relational data analysis and physical design and specifications. The tool runs on PC class machines, but links to mainframe data dictionaries. AUTO-MATE PLUS aids the developer in prototyping applications screens and report formats in the generation of IDMS/R database definitions. Graphics supported by the product include Data Flow Diagrams, Data Modeling, Online Control Structures (ADS/A) and Modular Dependency.

TOOL FACT SHEET

307

REF. #: 7 TOOL NAME: LEVERAGE

SOURCE: D. APPLETON COMPANY, INC. (DACOM)

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM 43XX OR 3XX	MVS TSO ISPF PLI			
DEC/VAX	VMS 4.4			

DESCRIPTION

LEVERAGE facilitates information systems planning, information systems development and data management. LEVERAGE is an automated language for defining information and systems requirements. The components of LEVERAGE include a Data Modeling Technique, Activity Modeling Technique, Team Management Technique and JANUS Modeling Software. The data modeling technique is similar to IDEF1, and the activity modeling technique is based on IDEFo.

TOOL FACT SHEET

308

REF. #: 8

TOOL NAME: PRECISE/IE

SOURCE: CDC

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
CYBER IBM PC/XT, PC/AT AND COMPATIBLES				

DESCRIPTION

The PRECISE/IE solution consists of a set of integrated models with a logical path from one level to the next. PRECISE/IE groups these models into four major planning and development phases.

PRECISE/INA	Information Engineering Needs Analysis
PRECISE/SIE	Strategic Information Engineering
PRECISE/TIE	Tactical Information Engineering
PRECISE/SDA	System and Data Architecture

By dividing the development process into separate phases, PRECISE/IE provides benefits to an organization even before the entire project is complete.

- The needs analysis conducted during PRECISE/INA looks at how an organization uses information and provides an overview of information processing requirements and a recommended plan of action. These recommendations consider short-term problems and long-term goals.
- PRECISE/SIE involves enterprise modeling for long-term planning, including a definition of the projects to be undertaken and an initial schedule for completion.

TOOL NAME: PRECISE/IE (cont'd.)

- PRECISE/TIE defines and documents detailed specifications for each of the projects identified during the strategic analysis phase.
- Finally, PRECISE/SDA relates the plans and specifications of the preceding PRECISE/IE phases to a specific, efficient technical and physical solution.

TOOL FACT SHEET

310

REF. #: 9 TOOL NAME: STRADIS/DRAW
SOURCE: MCDONNELL DOUGLAS INFORMATION SYSTEMS GROUP
FIRST RELEASE DATE: CURRENT RELEASE LEVEL:
APPROX. INSTALLATIONS: PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM 370	MVS/TSO OS/TSO VM/CMS			

DESCRIPTION

STRADIS/DRAW graphics software system allows for the creation and updating of two dimensional structured diagrams in a time sharing environment. System capabilities include interactive construction of Data Flow Diagrams, System Structure Charts and Design Data Flow Diagrams.

D17-01-09

TOOL FACT SHEET

311

REF. #: 10

TOOL NAME: SYSTEM DEVELOPER'S PRO KIT

SOURCE: MCDONNELL DOUGLAS INFORMATION SYSTEMS GROUP

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL: 2.0

APPROX. INSTALLATIONS:

PRICE RANGE: \$2,450 - PRO KIT ANALYST
\$600 - DFD DRAW
\$600 - SC DRAW

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/XT, IBM PC/AT, 3270 PC AND COMPATIBLES				

DESCRIPTION

SYSTEM DEVELOPER'S PRO KIT is made up of three separate products: PRO KIT ANALYST, DFD DRAW and SC DRAW. PRO KIT ANALYST is an automated graphics system used for drawing Data Flow Diagrams which generates its own data dictionary as the Data Flow Diagrams are created. DFD DRAW is a graphics software package which provides for the creation of Data Flow Diagrams. SC DRAW is a graphics product for the development and maintenance of Structure Charts.

TOOL FACT SHEET

312

REF. #: 11 TOOL NAME: DATA DESIGNER II

SOURCE: KNOWLEDGEWARE, INC.

FIRST RELEASE DATE: 1979

CURRENT RELEASE LEVEL: 2

APPROX. INSTALLATIONS: 150

PRICE RANGE: \$44,000 LESS DISCOUNT

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	MVS			
DEC/VAX				

DESCRIPTION

DATA DESIGNER II is a mainframe software tool to integrate the data needs of multiple users into a single logical data model and translate the model into physical design structures tailored to any type of DBMS or file access method. A variety of reporting and plotting options are available. The Data Model is represented in fourth normal form.

D17-01-11

TOOL FACT SHEET

313

REF. #: 12 TOOL NAME: CDEF

SOURCE: PRICE WATERHOUSE

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
------------	------------------------	------------------------	------------------------	---------------

DESCRIPTION

TOOL FACT SHEET

314

REF. #: 13 TOOL NAME: GAMMA

SOURCE: KNOWLEDGEWARE, INC.

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS: 40

PRICE RANGE: \$182,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	DOS/VSE VM/VMS MVS CMS/ISPF TSO/ISPF CICS	IMS/DC DL1 VSAM		

DESCRIPTION

GAMMA is an interactive application generator, which generates COBOL source code from system specifications. It generates up-to-date documentation, and acts as a project management tool. GAMMA development involves painting screen and report layouts. GAMMA translates these screens and reports into source code.

TOOL FACT SHEET

315

REF. #: 14 TOOL NAME: TELON

SOURCE: PANSOPHIC SYSTEMS, INC.

FIRST RELEASE DATE: 1981

CURRENT RELEASE LEVEL: 1.5

APPROX. INSTALLATIONS: 200

PRICE RANGE: \$130,000 - CICS/VSAM
\$300,000 - IMS/DC

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	CICS	IMS/DC VSAM		

DESCRIPTION

TELON is a high level design tool that addresses the entire development cycle. It is an interactive system for development of application systems for IMS/DC, CICS/OS and batch environments. It is composed of three software modules:

- Design Facility - Interactively captures design and programming specifications.
- Application Generator - Generates source code in COBOL or PL/1
- Modeling and Test Facility - Provides a simplified environment for modeling and assists in program testing

TELON's automated documentation feature provides the ability to summarize design information and track the use of databases.

TOOL FACT SHEET

316

REF. #: 15 TOOL NAME: TRANSFORM

SOURCE: TRANSFORM LOGIC CORPORATION

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE: \$225,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	CICS MVS	IMS DB/DC VSAM		

DESCRIPTION

TRANSFORM generates structured COBOL programs from non-procedural design input. A dictionary provides revision control over design elements and provides audit trails of their use. The Screen Painter Facility allow for quick specification of screen layouts. On-line system documentation provides a current, accurate and complete representation of the application programs.

D17-01-15

TOOL FACT SHEET

317

REF. #: 16 **TOOL NAME:** STRATEGIC SYSTEMS PLANNING
SOURCE: HOLLAND SYSTEMS CORPORATION
FIRST RELEASE DATE: 1981 **CURRENT RELEASE LEVEL:**
APPROX. INSTALLATIONS: **PRICE RANGE:** \$80,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
-------------------	--------------------------------	--------------------------------	--------------------------------	----------------------

DESCRIPTION

Strategic Systems Planning (SSP) helps develop an information strategy for the Enterprise. Using an Enterprise model and the analysis capabilities of SSP, an information strategy can be developed from a top down perspective. SSP can be used only with the Holland Strategic Systems Planning Methodology, although aspects of it are compatible with IBM's Business Systems Planning (BSP) and James Martin's Strategic Data Planning Methodology.

SSP supports the identification and analysis of common information needs among Enterprise processes, and employs function modeling and Affinity Analysis techniques.

TOOL FACT SHEET

318

REF. #: 17 **TOOL NAME:** LOGICAL DATABASE DESIGN
SOURCE: HOLLAND SYSTEMS CORPORATION
FIRST RELEASE DATE: 1983 **CURRENT RELEASE LEVEL:** 1
APPROX. INSTALLATIONS: **PRICE RANGE:**

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
-------------------	--------------------------------	--------------------------------	--------------------------------	----------------------

DESCRIPTION

Logical Database Design (LDD) is a mainframe software tool to integrate the data needs of multiple users into a single logical data model, and translate the model into physical design structures tailored to several types of DBMS's or file access methods. The logical data model produced is in fourth normal form. A variety of reporting and plotting options is available.

D17-01-17

TOOL FACT SHEET

319

REF. #: 18 TOOL NAME: PACBASE

SOURCE: CGI SYSTEMS, INC.

FIRST RELEASE DATE: 1972

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS: 500

PRICE RANGE: \$175,000 - \$300,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	DOS/CICS MVS/CICS MVS	IDMS AND OTHER CODASYL DBMSs		

HONEYWELL AND ASSOCIATED TP MONITORS AND DBMSs

SPERRY AND ASSOCIATED TP MONITORS AND DBMSs

DESCRIPTION

PACBASE supports the full application life cycle, and integrates and automates all phases of the application development and enhancement processes from system analysis through application generation and on-going maintenance. The system consists of five mainframe based components:

- The Specifications Dictionary, used to collect, store and maintain all system description information
- The Specifications Dictionary Manager, a production-quality database management system
- The Application Development Workbench, used to add, delete or modify application specifications in the dictionary
- The Program and Documentation Generators, used to create and document all of the operational components of the system
- The Table Manager, which organizes all table information without programming

TOOL NAME: PACBASE (cont'd.)

The Specifications Dictionary, the focal point for all PACBASE operations, is an information management system used to collect, store and maintain all information describing the system, including processing rules, data elements, data structure definition and relationships, documentation and screen report layouts.

The Specifications Dictionary Manager is a production-quality database management system. The Specifications Dictionary Manager's unique features include: journaling of dictionary entries, integrity controls that ensure database consistency and recovery in the event of a system failure, multi-user access to concurrent use of specifications from higher libraries, library management to provide effective data organization, and versioning for the formal control of application releases.

The Application Development Workbench provides integrated facilities for data modeling, database descriptions, prototyping, program design, report composition, screen design and documentation entry.

PACBASE Program Generators create all of the operational components of the system, including on-line and batch programs, screen maps, database descriptions and error messages. All processing logic is generated automatically from the specifications stored in the dictionary, so no programming exits need to be coded. And, if needed, special user routines can be added with the PACBASE Structured Code facility, or you can use COBOL directly.

The PACBASE Table Manager, PACTABLE, organizes and manages all table information without programming

TOOL FACT SHEET

321

REF. #: 19

TOOL NAME: APPLICATION PRODUCTIVITY SYSTEMS

SOURCE: SAGE SOFTWARE, INC.

FIRST RELEASE DATE: 1984

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS: 100

PRICE RANGE: \$100,000 - \$350,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	MVS	IMS DB/DC		
	MVS/XA	CICS/DL1		
	VM	DB2		
	PC/DOS	VSAM		
		IDMS		
		DATA COMM/DB		

DESCRIPTION

The APS Development Center offers integrated products which cover the entire development cycle. These products support the design, prototyping, development and maintenance of batch and online application systems.

The APS Development Center is a family comprised of five integrated product groups. These product groups support the full range of application requirements, from the early stages of system design through the implementation and maintenance of fully functional, performance-critical systems.

- The APS PC WORKSTATION product group focuses on the beginning of the development life cycle. These products enable both analysts and end users to draw, prototype and "test-drive" planned systems without programming. Development work is then uploaded to the mainframe host via the APS.
- The APS APPLICATION GENERATORS product group consists of mainframe-based products that accept input from the PC workstation and/or host-connected 3270 terminals. APS APPLICATION GENERATORS support application, screen, report, scenario, data structure and program painting as well as automatic program generation.

TOOL NAME: APS (cont'd.)

- The APS DB/DC product group automates the generation of programs for a variety of DB and TP environments. These products are tightly integrated with IBM's DB/DC offerings such as CICS, IMS/DC and ISPF data communications environments accessing DB2, DL1 and IMS databases and VSAM files. Additionally, APS DB products provide interfaces to IDMS and Datacomm/DB.
- The APS MAINTENANCE product group accepts existing database and screen code and populates the APS dictionary.
- The APS DICTIONARY group is the central repository for the APS Development Center. It provides support for the management of application system entities, and for JCL and document generation.

TOOL FACT SHEET

323

REF. #: 20

TOOL NAME: MAZDAMON

SOURCE: UCCEL

FIRST RELEASE DATE: 1983

CURRENT RELEASE LEVEL: 3.0

APPROX. INSTALLATIONS: 75

PRICE RANGE: \$30,000 - \$42,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	TSO/SPF	IMS CICS IDMS		VTAM
	MVS			

DESCRIPTION

MAZDAMON is a measurement device which monitors MVS networks and provides data for the effective management of network performance and precise measurement of end user response time.

FACT SHEET

324

REF. #: 21

TOOL NAME: DEFINE.IT

SOURCE/LOCATION: SOLION, INC.

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL: 1.0

APPROX. INSTALLATIONS: 15

PRICE RANGE: \$4750 LESS DISCOUNTS

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
MICRO-VAX	VMS	ORACLE	SQL	
CONVERGENT	UNIX	ORACLE	SQL	
IBM/XT + AT AND COMPATIBLES	MS/DOS	ORACLE	SQL	

DESCRIPTION

DEFINE.IT is a tool which supports use of the IDEF-1 Information Modeling approach. It contains adequate definition of Entity Class, Attribute Class and Relation Class characteristics to allow automatic generation of SQL build commands and the corresponding ORACLE Database to support system prototyping. The product disallows database generation if the model is incomplete or does not adhere to IDEF-1 rules. DEFINE.IT graphics conform to IDEF-1 notation.

TOOL FACT SHEET

325

REF. #: 22

TOOL NAME: INFORMATION MANAGEMENT FACILITY

SOURCE: BOOLE & BABBAGE

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM	IMS/VS			

DESCRIPTION

The IMF product family provides tools for analyzing, monitoring, managing, reporting and accounting for transaction processing in IMS/VS environments. Together, the seven products in the IMF product line provide a complete IMS/VS performance management system. They provide the functions of operational control, transaction flow analysis, bottleneck detection, service level surveillance, resource surveillance, transaction activity profiles and cost and usage allocation.

D17-01-22

TOOL FACT SHEET

326

REF. #: 23 TOOL NAME: XPF/COBOL

SOURCE: BOOLE & BABBAGE

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM				

DESCRIPTION

XPF/COBOL provides a structured testing environment in which a programmer can monitor and control the execution of COBOL programs. All testing and debugging can be accomplished interactively on terminals.

D17-01-23

TOOL FACT SHEET

327

REF. #: 24 TOOL NAME: TUTSIM

SOURCE: APPLIED I

FIRST RELEASE DATE: 1982 IN US

CURRENT RELEASE LEVEL: 5.02

APPROX. INSTALLATIONS: 1000 +

PRICE RANGE: \$495

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC/AT, PC/XT AND PC JR.				

DESCRIPTION

TUTSIM is a computer simulation program that permits a graphic representation of the behavior of linear or non-linear systems. TUTSIM constructs a block diagram simulation of a system, and supports execution of the simulation and evaluation of the results.

TOOL FACT SHEET

328

REF. #: 25

TOOL NAME: PAC II

SOURCE: AGS MANAGEMENT SYSTEMS

FIRST RELEASE DATE: 1979

CURRENT RELEASE LEVEL: 5.3 (IBM)

APPROX. INSTALLATIONS: 2000 (OVERALL)

PRICE RANGE: \$47,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
------------	------------------------	------------------------	------------------------	---------------

DESCRIPTION

PAC II is a comprehensive project management tool which provides the following functions:

- Planning
- Budgeting
- Project Simulation
- Scheduling
- Critical Path Analysis
- Skill Scheduling
- Resource Tracking
- Progress Tracking
- Project Monitoring
- Project Charge-back
- Change Management
- Reports
- Graphics
- Earned Value-performance Measurement

TOOL FACT SHEET

329

REF. #: 26 **TOOL NAME:** BLUE/60 THE DATA MODELER**SOURCE:** ABVENT**FIRST RELEASE DATE:****CURRENT RELEASE LEVEL:****APPROX. INSTALLATIONS:****PRICE RANGE:** \$1,495

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
MACINTOSH				

DESCRIPTION

BLUE/60 THE DATA MODELER is a tool which facilitates the creation and maintenance of:

- Data Models
- Entity-relation Diagrams
- Data Dictionary
- Automatic Version Number Creation
- Integrated Text Editors

TOOL FACT SHEET

330

REF. #: 27 TOOL NAME: LOGISCOPE

SOURCE: VERILOG

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
APOLLO	DOMAIN			
DEC	VMS			
IBM	VM/CMS, MVS XA			
SUN	UNIX			
CDC	NOS VE			
DPS8	MOLOCS			

DESCRIPTION

LOGISCOPE is a software quality analysis tool. It can be used during the basic software development tasks: design, coding, unit testing and integration testing.

LOGISCOPE was designed to provide:

- static analysis of the complexity of a program
- dynamic analysis of the test coverage

D17-01-27

TOOL FACT SHEET

331

REF. #: 28 TOOL NAME: ASA

SOURCE: VERILOG

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
VAX				
SUN				
APOLLO				

DESCRIPTION

ASA is a tool which provides for the creation of the following in the applications development process:

- State Machine Graphs
- State Transition Tables
- IDEF Diagrams
- Dictionary
- Test Suites
- Complexity Analysis
- Simulations
- Semantic Verification

D17-01-28

TOOL FACT SHEET

332

REF. #: 29 TOOL NAME: TAGS

SOURCE: TELEDYNE BROWN ENGINEERING

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL: 2.7

APPROX. INSTALLATIONS: 40 US

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
TBE 300 TBE 660 TBE 460 TBE 80	} NETWORK			
APOLLO				

DESCRIPTION

TAGS provides a facility for the automated definition, design, documentation, testing and maintenance of software systems. TAGS includes the IORL language and tools, such as:

- Storage and Retrieval Tool - manages design process entities
- Diagnostic Analyzer - locates construction errors
- Configuration Management - version control
- Simulation Compiler - executes IORL
- Analysis Library - allows engineering & management access
- Document Processor - integrates text & graphics

TOOL FACT SHEET

333

REF. #: 30

TOOL NAME: STATEMATE 1

SOURCE: AD CAD

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL: BETA SITE IN US

APPROX. INSTALLATIONS: 7

PRICE RANGE: \$10,000 - \$50,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
VAX	VMS			

DESCRIPTION

STATEMATE 1 is a comprehensive systems development tool which provides the following functions:

- Specification
- Design and analysis of real-time systems
- State Charts
- Activity Charts
- Module Charts
- Simulation Tools
- Management Tools
- Analysis Tools

TOOL FACT SHEET

334

REF. #: 31 TOOL NAME: COSTAR

SOURCE: SOFTSTAR SYSTEMS

FIRST RELEASE DATE: AUGUST 1986

CURRENT RELEASE LEVEL: 1.20

APPROX. INSTALLATIONS: 50

PRICE RANGE: \$400 - \$800

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
IBM PC	MS - DOS			

DESCRIPTION

COSTAR is a tool which provides for the implementation of Boehm's COCOMO estimating technique cost, effort & staffing for L-C phases.

TOOL FACT SHEET

335

REF. #: 32

TOOL NAME: REFINE

SOURCE: REASONING SYSTEMS

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL: 2.0

APPROX. INSTALLATIONS:

PRICE RANGE: \$100,000

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
SUN				

DESCRIPTION

REFINE provides a very high level specification language which can be compiled into an executable language; object-centered knowledge base.

D17-01-32

TOOL FACT SHEET

336

REF. #: 33

TOOL NAME: KEYONE

SOURCE: LPS

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
VAX	UNIX			
IBM PC/XT				
SUN				

DESCRIPTION

KEYONE is an applications development facility which includes the following tools:

- KEYDESIGN - incremental design analyzer, PDL based on ADA, PASCAL or C
- KEYDER - allows designer to derive implementation code, also replay derivation process for maintenance
- KEYFLE X - structured text editor
- KEYDOC - uses Knuth's WEB to intertwine code & documentation

TOOL FACT SHEET

337

REF. #: 34

TOOL NAME: ISTAR

SOURCE: IMPERIAL SOFTWARE TECHNOLOGY LIMITED

FIRST RELEASE DATE:

CURRENT RELEASE LEVEL:

APPROX. INSTALLATIONS:

PRICE RANGE:

HOST ENVIRONMENT(S)

MACHINE(S)	OPERATING SYSTEM(S)	DATABASE MANAGER(S)	COMMAND LANGUAGE(S)	COMMUNICATION
------------	------------------------	------------------------	------------------------	---------------

DESCRIPTION

ISTAR is an integrated, language independent, project support environment encompassing every aspect of software development - project management, data & configuration management and technical development.

D17-01-34

APPENDIX J

Tool / Life Cycle Usage / SDP-0 Class

ARTIFACT 7

TOOL/LIFE CYCLE USAGE/SDP-0

FUNCTIONALITY CLASS MATRIX

REV. 3
MAY 1987

ARTIFACT 7 - TOOL /LIFE CYCLE USAGE/SDP-0 FUNCTIONALITY CLASS MATRIX (TOOL MANUFACTURER VIEW)

340

TYPICAL LIFE-CYCLE USAGE

REF. #	TOOL NAME	PLANNING	ANALYSIS	DESIGN	CONSTRUCTION	OPERATION & MAINTENANCE	FUNCTION MODELING	INFORMATION MODELING	DYNAMICS MODELING	STATE TRANSITION MODELING	SIMULATION ANALYSIS	CODING & CLASSIFICATION	CONTROL CONTROL	PERFORM. ANALYSIS	LC ARTIFACT MANAGEMENT	DATABASE GENERATION	SOFTWARE GENERATION	ECONOMETRIC ANALYSIS	CLUSTER ANALYSIS	SURVEY SUPPORT	TEST CASE GENERATION
1	INFORMATION PLANNER	●	●				●													●	
2	ISS-THREE					●					✓			●							
3	INFORMATION ENGINEERING WORKBENCH		●	●			●	●	●	●											
4	EXCELLERATOR		●	●			●	●	●												
5	DESIGN AID		●	●	●		●	●									●				
6	AUTO-MATE PLUS		●	●	●		●	●								●	●				
7	LEVERAGE	●	●	●	●		●	●	●												
8	PRECISE/IE	●	●	●			●	●	●	●											
9	STRADIS/DRAW		●	●			●														
10	SYSTEM DEVELOPERS PRO KIT		●	●			●														
11	DATA DESIGNER II		●	●			●			●						●					
12	CDEF	●	●															●			
13	GAMMA			●	●	●				●			●		●	●	●				●
14	TELON		●	●	●	●				●			●		●	●	●				
15	TRANSFORM			●	●	●							●		●	●	●				
16	STRATEGIC SYSTEMS PLANNING	●	●																●		

DS6 01-01

ARTIFACT 7 - TOOL /LIFE CYCLE USAGE/SDP-0 FUNCTIONALITY CLASS MATRIX (TOOL MANUFACTURER VIEW)

341

FUNCTIONALITY PROVIDED

TYPICAL LIFE-CYCLE USAGE

REF. #	TOOL NAME	PLANNING	ANALYSIS	DESIGN	CONSTRUCTION	OPERATION & MAINTENANCE	FUNCTION MODELING	INFORMATION MODELING	DYNAMICS MODELING	STATE TRANSITION MODELING	SPECIFICATION REPRESENTATION	SIMULATION ANALYSIS	CODING & CLASSIFICATION	CONFG. CONTROL	PERFORM. ANALYSIS	LC ARTIFACT MANAGEMENT	DATABASE GENERATION	SOFTWARE GENERATION	ECONOMETRIC ANALYSIS	CLUSTER ANALYSIS	SURVEY SUPPORT	TEST CASE GENERATION
17	LOGICAL DATABASE DESIGN																					
18	PACBASE																					
19	APPLICATION PRODUCTIVITY SYSTEM																					
20	MAZDAMON																					
21	DEFINE.IT																					
22	INFORMATION MANAGEMENT FACILITY																					
23	XPFCOBOL																					
24	TUTSM																					
25	PAC II																					
26	BLUE/60 THE DATA MODELER																					
27	LOGISCOPE																					
28	ASA																					
29	TAGS																					
30	STATEMATE 1																					
31	COSTAR																					
32	REFINE																					

D56.01 02

ARTIFACT 7 - TOOL /LIFE CYCLE USAGE/SDP-0 FUNCTIONALITY CLASS MATRIX (TOOL MANUFACTURER VIEW)

TYPICAL LIFE-CYCLE USAGE

FUNCTIONALITY PROVIDED

REF. #	TOOL NAME	PLANNING	ANALYSIS	DESIGN	CONSTRUCTION	OPERATION & MAINTENANCE	FUNCTION MODELING	INFORMATION MODELING	DYNAMICS MODELING	STATE TRANSITION MODELING	SIMULATION ANALYSIS	CODING & CLASSIFICATION	CONFIG. CONTROL	PERFORM ANALYSIS	LC ARTIFACT MANAGEMENT	DATABASE GENERATION	SOFTWARE GENERATION	ECONOMETRIC ANALYSIS	CLUSTER ANALYSIS	SURVEY SUPPORT	TEST CASE GENERATION
33	KEYONE																				
34	ISTAR																				
35	MACH I																				
36	PROMOD																				
37	AMPLIFY CONTROL																				
38	SOFTWARE THROUGH PICTURES																				
39	INFORMATION ENGINEERING FACILITY (IEF)																				
40	SLAM II																				
41	DESIGN/I																				
42	SIMPLAN																				
43	DATA BASE PLUS																				
44	MANAGER FAMILY																				
45	EIFFEL																				
46	E R PACKAGE																				

END

DATE

FILMED

9-88

DTIC